- 8 -

Users

Introduction

There is a saying in IT that the perfect network doesn't have any pesky users. While it's true admin work would be much easier without users, it completely defeats the purpose of having a network at all. In fact, users should be a key focus point of your network because they are the ones that create the company and make sure it is creating revenue. Which in turn pays for the network and your salary; at least in most cases.

So, users are fundamental in your network and obviously, the topic of this chapter. In this chapter, we will discuss creating and managing users along with all the secondary configuration options that will benefit your users, network and your admin responsibilities.

While the Exchange Admin Center offers a great deal of configuration options regarding the creation, management, etc. of users, you will hopefully see the major benefits of PowerShell when requiring bulk creation and changes. Knowing what is possible with Exchange Management Shell (EMS) might have some impact on how best to provision your users, your use of attributes and other conventions. It might be prudent to review those practices.

Types of Objects

There are different types of objects that this chapter will address:

- User Mailbox
- Mail Enabled User
- Mail Contact
- Resource Mailbox
- Archive Mailbox
- Public Folder Mailbox
- Shared Mailbox
- Remote Mailbox

A **User Mailbox** is an AD user with a mailbox. This is different from a Mail User that can log in, but has no mailbox, and will only forward to another email address. A Mail Contact is an object that represents an email address in another environment, but is not security enabled and therefore cannot login.

Resource Mailboxes have a disabled user account, like a User Mailbox they also have a calendar, but it is designed for resource reservations like rooms and/or equipment. There are two types of resource mailboxes: Room and Equipment. These have a few different attributes and additional features to support planning meetings or equipment information.

A **Shared Mailbox** has a disabled user account. The idea is that normal user mailboxes get permission to access email and send as that mailbox, with all the data stored in that mailbox and not across different mailboxes. Useful for general email addresses like info@company.com etc., in which several people require access and send permissions.

With Modern Public Folders introduced in Exchange Server 2013, the infrastructure has changed radically and Public Folder data is no longer stored in a separate database, but in **Public Folder Mailboxes** in mailbox databases. This change was carried forth in Exchange Online all the while the user experience has not changed.

An **Archive Mailbox** is an additional mailbox linked to the user's primary mailbox, with the distinct difference that the Archive mailbox is only available via Outlook Desktop (ProPlus) and Outlook Web App (OWA) edition when connected to Exchange. This means that no offline access is available per design, it is meant primarily as a PST replacement.

Creating Users

To explain the intricacies of creating users who will be able to send and receive email, we should look a little into how Azure Active Directory (AAD) works and how Exchange Online leverages it.

Mailbox or Mail Enabled User

There are two kinds of users possible, mailbox or mail enabled. The first is a user account that can be authenticated by the Azure Active Directory and has a mailbox connected to it. This means the user can send and receive email, manage calendars, contacts, etc. All that information is stored in Exchange Online.

A mail enabled user (or mail user for short) can also be authenticated by the Azure Active Directory but does not have a mailbox. Their user object does have an email address and a forwarding address, most likely to a mailbox in another environment (i.e partner company). If anyone sends an email to this user, Exchange will forward the email to the forwarding address. For instance, if you are an IT consultant with multiple customers, it's reasonable to not have to maintain multiple mailboxes. This way users of each environment can find you in the Address List and send email, you on the other hand will get all email in one mailbox depending on your forwarding address. Nice to know: that will also limit the need for Exchange Online Licenses.

New Mailbox

You can directly create a new mailbox without the need to create an Azure Active Directory user first, it will be automatically created. However, the options available to you are mostly limited to things related to Exchange, for instance a home path cannot be configured in the same action. You will probably need to configure the user with MSOL cmdlets if so required.

Note: If your environment is synced, creating users and mailboxes with New-MSOLUser is not a recommended process. Instead the objects (user account and remote mailbox) should be initiated from your on-premises Exchange and Active Directory servers. If your environment is not synced and is cloud-only, then using New-MSOLUser is the appropriate course of action as this will be the only way to create new objects in Exchange Online.

Use the following command to create a MSOL User object as the first step to creating a mailbox, we need to use the New-MSOLUser cmdlet:

```
New-MsolUser -UserPrincipalName Dom.Rigel@Contoso.Com -DisplayName "Dom Rigel" -FirstName
 "Dom" -LastName "Rigel" -UsageLocation "US"
```

Note: When using spaces in the Name field, you are required to use quotation marks if there is a space. The UserPrincipalName should obviously be valid for the Domain (you might need to add UPN suffixes) and preferably should correspond to the primary SMTP address the account will be using.

You can define the password via a prompt:

```
$SecurePassword = Read-Host -Prompt "Enter password" -AsSecureString
```

or a pre-determined value:

```
$PlainPassword = "Th1sSho4ldB3Secr3t"
$SecurePassword = $PlainPassword | ConvertTo-SecureString -AsPlainText -Force
```

In both cases the passwords must adhere to the password policy in place. The first method is fine for single changes. The latter is ideal for bulk additions of mailboxes. Obviously, you can also define a randomly generated unique password for each new mailbox, which is from a security perspective preferable.

Once a user is created, it needs to be licensed for Exchange Online. See Chapter 6 for those details. Once the user is assigned a license, it does take a period of time for the mailbox to be created in Exchange Online. Once it is, you will be able to work with it in the Exchange Admin Center or Exchange PowerShell. To define FirstName, LastName etc. at account creation:

```
Set-MsolUser Dom. Rigel@Contoso.Com -FirstName Dom -LastName Rigel
```

In order to configure additional parameters for a mailbox we can use the 'Set-Mailbox' cmdlet. This cmdlet has a few parameters we can use to configure a mailbox:

PrimarySmtpAddress: Which defines the primary SMTP address or reply address for that mailbox. Do note that when using this parameter, the Email Address Policy (EAP) setting EmailAddressPolicyEnabled is set to $False which means no EAP is applied on this account. This can be useful if you don't want this account to have all the SMTP addresses applied from an EAP, for instance with Shared Mailboxes. Or this mailbox will be used for a very specific purposes, requiring only the set address. It is worth noting that creating and utilizing EAPs requires an on-premises Exchange server as this option is not available in Exchange Online.
AccountDisabled: When creating the mailbox and logon account, security policies might dictate you to disable the Active Directory account until it's ready for use (maybe additional security settings are required) or when the actual user is allowed to use it. For these circumstances, you can use this switch, but no value is required (i.e. $False isn't needed).
Name, DisplayName, FirstName, LastName are all values that have a special relationship together. These values are used by other users or even admins to identify the correct mailbox to the real-world user. Especially in large organizations it is prudent to have a good naming convention in place, also planning for all deviations that will happen. No naming convention will incorporate every possible situation, especially if your users have very different cultural naming standards and practices.

FirstName and LastName were already discussed, but it still important to point out that these values can be used in your Email Address Policy. So, even though they are not mandatory, it might help with your Email Address Policies or help your users find the correct person within Exchange Online.

You can further specify the user's Office and Phone parameters. These are, by default, visible in Address Lists etc., so make sure that privacy regulations in your country/region are followed. Note that you can filter based on Office locations when using RecipientFilter with many Exchange cmdlets.

Furthermore, not even all Exchange related values can be set when creating the mailbox, it is highly likely that you must use other cmdlets to completely configure the mailbox account to your organizations requirements and/or liking.

User Policies: There are several types of policies available in Exchange, you can set specific (custom) policies with the following aptly named policy parameters:

- ActiveSyncMailboxPolicy
- AddressBookPolicy
- RetentionPolicy
- RoleAssignmentPolicy
- SharingPolicy

Enabling Mailboxes

If your user(s) already have an Active Directory account, some parameters are already configured via other means. This way you only must concentrate on Exchange specific attributes and thus cmdlet parameters. You can mailbox enable a user with at least these parameters:

```
Enable-Mailbox -Identity Sjon.Lont
```

In this case Identity can be the Name, Display Name or other types of values, that can uniquely identify the target user account. Note that when not specifying other parameters the default values are used, the same when using the New-Mailbox cmdlets.

The cmdlet further behaves the same as the New-Mailbox cmdlet, with the distinct difference the AD account and the mandatory values are already provided.

Enabling an Archive Mailbox

You can enable the Archive Mailbox on an existing mailbox user with an archive switch:

```
Enable-Mailbox Dom.Rigel -Archive -ArchiveName "Dom Rigel Archive"
```

The ArchiveName specifies the name that identifies the Archive mailbox, otherwise the default naming is used, which is "In-Place Archive – " before the mailbox display name.

You can also create a new mailbox immediately with an archive by adding the -Archive switch to New-Mailbox or Enable-Mailbox.

The default archive quota and archive quota warning are 100 GB and 100 GB. These values cannot be changed in either the Admin Center or via PowerShell. However, we can adjust the archive and make it into an AutoExpanding archive if we have the proper licenses. This can be done with this command:

```
Enable-Mailbox Dom.Rigel -AutoExpandingArchive
```

Do not forget to apply the appropriate Retention Policy, which can be specified when Archive enabling the user with the -RetentionPolicy parameter.

New Mail Contacts

Mail enabled contacts are a way to create entries in the Global Address List that users can use to email often used addresses outside of your environment. For instance, if you have a Shared Service Desk supplier, you can create a mail contact with a recognizable name and an internal email address which also contains a forwarding address. To create a contact:

```
New-MailContact -Name "Richard Deck" -ExternalEmailAddress R.Deck@Outlook.Com
```

Note: The contact will get an SMTP address according Email Address Policy settings, however the ExternalEmailAddress is the primary address and all email sent to the contact will be forwarded to the external address.

To delete a contact:

```
Remove-MailContact "Richard Deck"
```

Deleting Users

There are two options: deleting the mailbox or deleting the user account including the mailbox. It depends on your own requirements and situations which of the two options is valid.

To remove the mailbox and NOT the user:

```
Disable-Mailbox -Identity Dom.Rigel -PermanentlyDisable
```

Note: This only works if the user does not have an assigned Exchange License. In order to remove a mailbox without removing the user, the license for Exchange would have to be removed. See Chapter 6 for how to change licenses for users.

To remove the mailbox AND the user:

```
Remove-Mailbox -Identity Dom.Rigel -Confirm:$False
```

Alternatively, it's also possible to change the type of a User mailbox to Shared to keep the data and email flow available. See later in this chapter on how to do this.

Modifying Users

There is one constant, and that is that things change. This is definitely the case for users. A lot can be changed via Exchange PowerShell and it's probable that a lot of settings are never changed or will require being changed.

But when it comes to modifying mailbox users, there are several things to consider. Most importantly, there is no single cmdlet that can modify everything on a mailbox. You must use the correct cmdlet for the required changes you want to make.

For an overview of all the attributes that (might) be subject to any modification, see the Reporting section later in this chapter. In that overview the Get-* cmdlets are used, but obviously to change the attributes you should use the Set-* variant or in some cases (like permissions) the option to use Add-* or Remove-* is also an option.

It's not the goal of this book to review every possible modification available, we will show what we feel are the most important and common modifications.

User

The user object is where it all starts, whether it has a mailbox or is only mail-enabled. You can change the UPN of a user in the cloud, or maybe change the Display Name. To change a UserPrincipalName, there is a command called 'Set-MsolUserPrincipalName' like so:

```
Set-MsolUserPrincipalName -UserPrincipalName user@contoso.com -NewUserPrincipalName
 johns@contoso.com
```

Note: Don't forget to connect to the MSOL service with either Connect-MSOLService or Connect-EXOPSSession (MultiFactor) before running these cmdlets. Also, changing UPNs might have adverse effects and should be researched before changing.

There are a few other cmdlets that can be used to manipulate user accounts and we can find them again with the Get-Command technique:

```
Get-Command *MsolUser*
```

Which provides us with a short list of cmdlets:

```
Get-MsolUser                Get-MsolUserByStrongAuthentication     Get-MsolUserRole
New-MsolUser                Remove-MsolUser                        Restore-MsolUser
Set-MsolUser                Set-MsolUserLicense                    Set-MsolUserPassword
Set-MsolUserPrincipalName
```

Mailbox

There are several cmdlets that configure options on a (user) mailbox, most of those features are set with Set-Mailbox. There are other cmdlets that set other very specific settings, so if this cmdlet doesn't provide what you want to change you might have to use another cmdlet.

Why not in one cmdlet? Some of the settings control specific user settings that a user should have access to. Because of that everything is controlled in one way or another with PowerShell and Role Based Access Control (RBAC), it's sometimes easier to have a separate cmdlet for specific settings that are also configurable by users. It makes it easier to control those permissions (via Role Assignments with RBAC).

Settings on the mailbox include some email flow control such as addresses, forwarding or size/delivery restrictions, storage or quota settings and some policies. Others include junk email handling, OWA configuration (including features other than what is set via OWA Mailbox policies) and regional settings.

We will discuss some cmdlets in more detail below, in a per cmdlet way instead of a per scenario way.

Set-Mailbox

One example to change the email flow settings is to set a forwarding address to another user:

```
Set-Mailbox -Identity Gene.Ricks@Contoso.Com -DeliverToMailboxAndForward:$True
 -ForwardingAddress Ann.Ples@Contoso.Com
```

Above we set this to an SMTP address, for which there must be a matching mail object for it; a mailbox, mail user or mail contact. If you needed to forward a message to an external SMTP address the ForwardingSmtpAddress property would be set instead.

With delivery restrictions, you can control what email is accepted or not.

```
Set-Mailbox -Identity Gene.Ricks@Contoso.Com -RequireSenderAuthenticationEnabled:$True
 -AcceptMessagesOnlyFromSendersOrMembers @('Ann.Ples@Contoso.Com')
 -RejectMessagesFromSendersOrMembers @('Mike.Soft@Contoso.Com')
```

With RequireSenderAuthenticationEnabled only accounts in your Exchange Online Tenant can email this mailbox (this is enabled for distribution groups by default). You can also configure users or groups to be accepted or rejected explicitly, in this example email from Ann Ples is accepted and from Mike Soft is rejected. Note that those are multi-valued properties.
Another setting is the mailbox quotas, mainly the IssueWarningQuota, ProhibitSendQuota and ProhibitSendReceiveQuota settings. There are also quotas when using auditing and Litigation/In-Place hold, but the principle is the same. The big difference is that most mailboxes will use the default Database quota settings, but in case you need to override those settings you have to set them on the mailbox:

```
Set-Mailbox -Identity Gene.Ricks@Contoso.Com -IssueWarningQuota '10737418240'
 -ProhibitSendQuota '11811160064' -ProhibitSendReceiveQuota '12884901888'
```

In this example the mailbox quotas are respectively 10GB, 11GB and 12GB, the normal input is in MB (megabytes) however you can explicitly state whether you use MB or GB etc.:

```
Set-Mailbox -Identity Gene.Ricks@Contoso.Com -IssueWarningQuota 10GB
```

If you require to retain deleted items longer than the default 14 days, then we need to change the mailbox default:

```
Set-Mailbox -Identity Gene.Ricks@Contoso.Com -UseDatabaseRetentionDefaults:$false
 -RetainDeletedItemsFor '30'
```

In this example the database retention defaults are disabled; the deleted items are retained for user recovery for 31 days. There is a limit in Exchange Online for the RetainDeletedItemsFor value and that is 30. If we try to set it greater than 30, then we get an error like so:

```
PS C:\> Set-Mailbox -Identity Damian -UseDatabaseRetentionDefaults:$false -RetainDeletedItemsFor '32'
The operation on mailbox "damian" failed because it's out of the current user's write scope. The value of properties
'RetainDeletedItemsFor' exceeds the maximum allowed for user 'damian' with license 'BPOS_S_Enterprise'.
    + CategoryInfo          : InvalidOperation: (damian:ADObjectId) [Set-Mailbox], InvalidOperationException
    + FullyQualifiedErrorId : [Server=CY4PR13MB1350,RequestId=27b218de-57d6-45b0-8494-1bbabdcebbbc,TimeStamp=12/7/2017
   3:11:38 AM] [FailureCategory=Cmdlet-InvalidOperationException] 4D104F39,Microsoft.Exchange.Management.RecipientTa
  sks.SetMailbox
    + PSComputerName        : ps.outlook.com
```

For setting additional SMTP addresses see example Adding/Removing an email address later in this chapter.

Set-MailboxAutoReplyConfiguration
Configures Out of Office (OOF) replies, including scheduling, inside and outside organization message. Basically, every possible setting the user can set. See Enabling and Configure Out of Office settings by the admin for an example.

Set-MailboxJunkEmailConfiguration
Configure the User Junk folder with specifics in addition to what the user has configured via OWA/Outlook.

```
Set-MailboxJunkEmailConfiguration -Identity Gene.Ricks@Contoso.Com
 -TrustedSendersAndDomains fabrikam.com
```

The above will add the fabrikam.com domain as a trusted sender and Exchange will handle those domains differently (however if you have valid spam filtering software, their settings probably take precedence).

Set-MailboxRegionalConfiguration
Configures regional settings on a specific mailbox, such as timezone, date format, language etc.. Users will be prompted the first time they log in OWA or it will be configured depending on the client. However, as an admin you can provision these settings.

See Setting Regional setting in this chapter for an example.

OWA

There are some settings specifically for OWA. The user can change these settings, but as in other similar examples it might be required to provision some settings for users.

Set-MailboxMessageConfiguration
Configures the behavior of OWA for a specific mailbox. For instance the automatic addition of a signature, always show the 'From:' field when composing messages, conversation order and whether ReplyAll is the default response:

```
Set-MailboxMessageConfiguration -Identity Gene.Ricks@Contoso.Com -AutoAddSignature $True
 -AlwaysShowFrom $True -ConversationSortOrder ChronologicalNewestOnTop
 -IsReplyAllTheDefaultResponse $False
```

Set-MailboxSpellingConfiguration
Set the spelling language in OWA, force check before sending the email and whether to ignore uppercase and mixed digits:

```
Set-MailboxSpellingConfiguration -Identity Gene.Ricks@Contoso.Com -CheckBeforeSend $True
 -IgnoreUpperCase $True -IgnoreMixedDigits $True -DictionaryLanguage Dutch
```

Calendar

Calendar settings can be changed to affect the way calendar invites are processed or to set time zones for instance. While you can configure calendar settings for user mailboxes during provisioning, you will likely have to perform these actions more often for Room and Equipment Mailboxes as users can change most of these settings themselves.

Set-MailboxCalendarConfiguration

Can change calendar configurations and is available to the user, but also the admins so they can provision certain settings for the users. Such as WorkDays/WorkingHours, the first week of the year, timezones and such. Some customization are for OWA only as Outlook (or other clients) have their own settings that supersede these.

```
Set-MailboxCalendarConfiguration -Identity Gene.Ricks@Contoso.Com –WeekStartDay Monday
```

Sets the first day of the week to Monday, instead of the default Sunday.

Set-MailboxCalendarFolder

This cmdlet is only relevant when sharing a calendar with a federated Exchange organization or when Internet Publishing is allowed. You can reset the published URLs, change the date range of what is published and disable the sharing. You can only do this for your own mailbox, unless you change the Role Assignment.

```
Set-MailboxCalendarFolder administrator:\Calendar -PublishEnabled $True -DetailLevel
 Limited
```

Set-CalendarProcessing

The cmdlet Set-CalenderProcessing configures the way Exchange will handle meeting requests. As previously stated, users can configure these settings themselves and some settings are not relevant for user mailboxes. However, they are for Room and Equipment mailboxes which can turn them into automatic booking systems. You can use the same principal for inactive mailboxes, previously owned by users and setting to refuse every meeting request.

```
Set-CalendarProcessing -Identity Auditorium -ProcessExternalMeetingMessages $True
 -AutomateProcessing AutoAccept -AddOrganizerToSubject $True -AddAdditionalResponse $True
 -AdditionalResponse "Your request has been accepted."
```

This example configures the Room mailbox Auditorium to process External meeting requests (coming from outside of the Exchange organization), automatically accepts the requests, changes the Subject to the name of the organizer and will reply with a customized response to the organizer.

You can set additional options like whether users can set a reoccurring meeting, maximum meeting duration and delegates that have to give approval.

Client Access

All client access related settings are performed with Set-CASMailbox. You can disable/enable and configure specific protocols, such as IMAP/POP or OWA, ActiveSync and Exchange Web Services (EWS). Basically, everything mailbox client connection related (with the exception of SMTP) can be configured.

```
Set-CASMailbox -Identity Gene.Ricks@Contoso.Com -PopEnabled $False -ImapEnabled $False
 -EwsAllowEntourage $False -ActiveSyncEnabled $False
```

In this example, POP, IMAP and ActiveSync are disabled and EWS Entourage support (an Outlook for MacOS predecessor) is not allowed. Note that IMAP and POP are default enabled, but the service is by default disabled on every Exchange server. Thus, if an application or user requires either one of the protocols the services must be enabled and started. It's a best practice to disable these protocols or to not publish the ports to the Internet as a way of increasing security.

Policies

Policies are an easy way to ensure users get the right configuration and is preferable to changing each specific user. There are several policies available:

- OWA Mailbox
- Retention
- RoleAssignmentPolicy
- SharingPolicy
- Mobile Device

OWA Mailbox policies regulate the Outlook Web App capabilities available to the user, the default has every feature enabled. For instance, Offline Mode is an often-disabled feature in the default policy or other custom policies.

Retention policies give users and admins the option to regulate the retention of items in their mailbox or specific folders. When the mailbox is Archive enabled Retention policies (with the "Move to Archive") are commonly used, but an Archive mailbox is not required for their use.

Role assignments are part of Role Based Access Control (RBAC), the security model within Exchange. These policies regulate what users can carry out what actions on what objects, such as updating a distribution group for instance.

Sharing policies regulate sharing of calendar information within federated Exchange organizations or via Internet Calendar Publishing.

Mobile Device policies configure the security settings and features on connected mobile devices, via Exchange ActiveSync or the Outlook for iOS/Android app. Most commonly a mandatory PIN is set via these policies.

Obviously to assign or to change policies, the policies must exist. Assigning or changing the assigned policy on a mailbox is done via the Set-Mailbox or Set-CASMailbox cmdlet:

OWA Mailbox Policy:

```
Set-CASMailbox -Identity Gene.Ricks@Contoso.Com -OwaMailboxPolicy NoOfflineOWA
```

Retention Policy:

```
Set-Mailbox -Identity Gene.Ricks@Contoso.Com -RetentionPolicy AutoCleanDeletedItems
```

Role Assignment Policy:

```
Set-Mailbox -Identity Gene.Ricks@Contoso.Com -RoleAssignmentPolicy EditSubsetGroups
```

Sharing Policy:

```
Set-Mailbox -Identity Gene.Ricks@Contoso.Com -SharingPolicy InternetSharing
```

ActiveSync Mailbox Policy:

```
Set-CASMailbox -Identity Gene.Ricks@Contoso.Com -ActiveSyncMailboxPolicy HighSecurity
```

See **Chapter 9** for more information on managing non-user objects. For Mobile Device policies check **Chapter 15**.

Permissions

These are different levels of permissions possible on Exchange mailboxes:

- Full Access
- Send As
- Send On Behalf
- Folder Permissions

To add Full Access permissions, use Add-MailboxPermission:

```
Add-MailboxPermission -user Mike.Soft -identity Ann.Ples -AccessRights FullAccess
 -InheritanceType All -Automapping $False
```

In this case Mike Soft will be granted full access on Ann Ples' mailbox, additionally this permission will be granted to all folders within the mailbox. The setting Automapping controls whether Ann's mailbox is automatically added in Mike's Outlook (via AutoDiscover), in this case by setting it to $false it will not. When the Automapping feature is not configured it is default True (which is also the case when using the Exchange Admin Center). This will not grant Send-As permissions, that is actually a recipient based permission and can be set via:

```
Add-RecipientPermission -Identity Ann.Ples -Trustee Mike.Soft -AccessRights 'Send As'
```

In this example user Mike has been granted Send As permissions to Ann's mailbox. Do note that Mike has to change the 'From:' value in Outlook to Ann's email address. In cases where it is required that the actual sender is still visible, Send on Behalf is the best option. This must be configured with the Set-Mailbox cmdlet:

```
Set-Mailbox –Identity Ann.Ples -GrantSendOnBehalfTo Mike.Soft
```

In this example Mike has been granted Send on Behalf permissions. As with Send-As, Mike must change the 'From:' value in Outlook to make use of this permission. However, the recipient will now see the actual sender even if replies are sent back to the main mailbox (Ann's). In some cases, Full Access is too broad therefore it is good to be able to set permissions on specific folders. Folder Permissions are set via the user itself in Outlook or OWA, but admins can use:

```
Add-MailboxFolderPermission -Identity Ann.Ples:\Inbox -User Mike.Soft -AccessRights Owner
```

In this example, Mike gets Owner permissions on the Inbox folder inside Ann's Mailbox. There are quite a lot of different permissions possible, be sure to read up on them at Microsoft Docs. Note that the Calendar folder has two additional permission roles specifically for availability visibility. Follow-up reading.

In this example, the Add-MailboxFolderPermission was used which adds permissions and lets previously set (not inherited) permissions as is. Use the Set-MailboxFolderPermission to edit previously assigned permissions and Remove-MailboxFolderPermissions to remove permissions.

```
[PS] C:\>Add-MailboxFolderPermission -Identity Ann.Ples:\Inbox -User GeneRicks -AccessRights Owner

FolderName            User                 AccessRights      SharingPermissionFlags
----------            ----                 ------------      ----------------------
Inbox                 Gene Ricks           {Owner}


[PS] C:\>Set-MailboxFolderPermission -Identity Ann.Ples:\Inbox -User GeneRicks -AccessRights Editor
[PS] C:\>Get-MailboxFolderPermission -Identity Ann.Ples:\Inbox

FolderName            User                 AccessRights      SharingPermissionFlags
----------            ----                 ------------      ----------------------
Inbox                 Default              {None}
Inbox                 Anonymous            {None}
Inbox                 Gene Ricks           {Editor}
```

```
[PS] C:\>Remove-MailboxFolderPermission -Identity Ann.Ples:\Inbox -User GeneRicks

 onfirm
Are you sure you want to perform this action?
Removing mailbox folder permission on "Ann.Ples:\Inbox" for user "Gene Ricks".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [?] Help (default is "Y"):
[PS] C:\>
[PS] C:\>Get-MailboxFolderPermission -Identity Ann.Ples:\Inbox

FolderName           User                 AccessRights     SharingPermissionFlags
----------           ----                 ------------     ----------------------
Inbox                Default              {None}
Inbox                Anonymous            {None}
```

Note: Well-Known folders (like Inbox, Calendar, Sent Items etc.) will change with regional settings set by the user (via OWA) or by language settings of Outlook when first connecting to their mailbox. This might pose a challenge if you want to automate specific settings on those Well-Known folders. Luckily the FolderType is a constant and that value will tell you what kind of folder it is.

Custom made folders (a second calendar for instance) have the folder type of "User Created". Use the following PowerShell one-liner in order to find the specific name of the Well-Known Calendar folder:

```
Get-ExOMailbox <Mailbox>| Get-ExOMailboxFolderStatistics|Where {$_.FolderType -eq
 "Calendar"}
```

The value of the FolderType can be Inbox, Contacts, Sent Items, Deleted Items etc.. You can list this for a specific mailbox with:

```
Get-ExOMailbox <mailbox> | Get-ExOMailboxFolderStatistics| Select FolderType
```

Often Requested Changes

Below is a long list of often performed changes.

Enabling and configure Out of Office settings by the admin

Even with all the options available to a user to configure the Out of Office (OOF, which is an abbreviation for Out of Facility, harking back to early Exchange years when its predecessor was used internally), you might get requests to set this. Luckily this is relatively easily done:

```
Set-MailboxAutoReplyConfiguration -Identity Ann.Ples -AutoReplyState Enabled
 -InternalMessage "I'm currently out of office."
```

This is the simplest configuration, there are options to set a message for external users (i.e. not in the Exchange organization), setting a time period when the OOF status has to be enabled, automatically declining meeting request etc. Basically, every setting available to the user, when using a recent version of Outlook or OWA. However, for an admin this example will probably be sufficient in most cases.

Regional Setting

In some cases you want to set regional settings for a user, so that the user has an even more fluid first logon experience and isn't bothered by questions about language etc.. Especially valid if your organization is set in a single language region etc.. You can set the specific regional settings with:

```
Set-MailboxRegionalConfiguration -Identity "Hans de Vries" -Language nl-nl -DateFormat
 "dd-MM-yy" -LocalizeDefaultFolderName -TimeZone "W. Europe Standard Time"
```

In this example the user will have Netherlands Dutch language settings in OWA and in Outlook, the LocalizeDefaultFolderName parameter will change the well-known folders like Inbox and Calendar to the localized versions (respectively 'Postvak IN' and 'Kalender' in this case). The latter could be important when you have scripts to set Calendar

folder permissions or require migrating to a non-Exchange environment via PST. Furthermore, the date format has been set to correspond with the region and the time zone has been set to West Europe.

Adding/Removing an Email Address
Even with Email Address Policies it is possible that the naming convention doesn't provide the required SMTP address. Or the account requires additional SMTP addresses or the user changed his or her name.

Note: Don't forget that modifying properties of an object while connected to Exchange Online PowerShell is only supported for cloud only objects.

You can add email addresses with the Set-Mailbox cmdlet with the EmailAddresses parameter. However, if you use this parameter the value will replace all of the configured addresses (only to be added again by the Email Address Policy). Therefore you need to use a little different syntax:

```
Set-Mailbox Ann.Ples -EmailAddresses @{Add="smtp:Ann.Ples@Contoso.Com"}
```

In this example the Ann.Ples@Contoso.Com address is added to other addresses on the Ann Ples mailbox. Note the small caps type "smtp". If you change this to capital letters, this will become the Primary SMTP address. However, this could be overruled by any active Email Address Policy (from an on-premises Exchange Server).

Removing an email address is achieved by using Remove instead of Add:

```
Set-Mailbox Ann.Ples -EmailAddresses @{Remove="smtp:Ann.Ples@Contoso.Com"}
```

In both cases, other SMTP addresses configured on the mailbox are not removed.

Converting Mailbox Types

There are times you might have to change the type of the (user) object to another; types being user, resource or shared. Sometimes, converting a mail user to a full mailbox enabled user is required or when mergers have been completed Linked mailboxes might have to be converted to user mailboxes. This quickly summarizes the options and some of the things you should consider.

Converting Mail User to User Mailbox

There are situations that might require you to convert a Mail User to a Mailbox User.

In order to convert a Mail User into a Mailbox, you will need to assign an Exchange Online License. First, you will need to create the Mail User in Office 365:

```
PS C:\> New-MailUser -Name 'Pete Blanket' -ExternalEmailAddress PBlanket@Outlook.Com -MicrosoftOnlineServicesID pbla
nket@OnlineExchangeBook.onmicrosoft.com

cmdlet New-MailUser at command pipeline position 1
Supply values for the following parameters:
Password: ********

Name                                    RecipientType
----                                    -------------
Pete Blanket                            MailUser
```

Converting User Mailbox to Shared Mailbox

This might be useful when a person leaves the organization, but you are required to keep the data intact due to legal and/or compliance regulations. By converting the mailbox from user mailbox to shared, you disable the AD account (lowering the risk of breaches), give access to others within the company and you are still able to keep the mail flow intact.

Note: Shared mailboxes do not require an Exchange Online license, unless a Legal Hold is put in place or any other feature is used on the mailbox that requires a license.

You can convert mailbox types with the Set-Mailbox cmdlet:

```
Set-Mailbox -Identity PeteBlanket -Type Shared
```

You can change the type of mailboxes to Resources (Room, Equipment), Shared or User Mailboxes in this way with the Set-Mailbox cmdlet. Valid values are: 'Regular', 'Room', 'Equipment' and 'Shared'. When changing the type of the mailbox, do not forget that additional configuration specifics for the new mailbox type is required depending on your organizational needs.

Reporting

In this section, we will give some attention towards reporting on user mailboxes. This is discussed in more depth in Chapter 18 - Reporting, but there are some specifics that warrant a mention in this chapter.

General Remarks

It's prudent to check with every update if there are any new commands or new attributes exposed in the Get cmdlets. Especially when new features are added, you'd expect to find some way of configuring those features. However, as Exchange is developed with Office 365 (or specifically Exchange Online) in mind you might encounter attributes that are of no use in the cloud (or vice versa). You can ignore those.

When you have a lot of objects, do not forget to add the -ResultSize parameter to your cmdlet preferably with "Unlimited" as a value, otherwise only 1,000 objects are returned. You will get a warning, but within a script you might miss that and it could result in incomplete processing or reporting of your environment. For testing purposes, you could use this to limit the number of objects returned and thus speed up your script.

```
WARNING: By default, only the first 1000 items are returned. Use the ResultSize parameter to
specify the number of items returned. To return all items, specify "-ResultSize Unlimited". Be
aware that, depending on the actual number of items, returning all items can take a long time and
consume a large amount of memory. Also, we don't recommend storing the results in a variable.
Instead, pipe the results to another task or script to perform batch changes.
```

Cmdlets

Let's see the relevant cmdlets, what kind of information they reveal and in some instances, some extra useful information. Do note that some cmdlets are not that obvious. Check the screenshots for some formatting suggestions, some -Identity fields are sometimes a bit more complex than just adding user identity values. Also, some cmdlets show more interesting information when you pipe the cmdlet to Format-List (FL in short), this has been used in the examples but is not required when using a script (as PowerShell returns objects not text).

Get-MsolUser

Lists attributes such as phone, address names, etc. from Active Directory users, whether they are mail- or mailbox enabled or not. The focus is the Active Directory user object rather than the mailbox.

```
Get-MsolUser -Identity Damian@OnlineExchangeBook.OnMicrosoft.Com | Fl
```

```
ExtensionData                       : System.Runtime.Serialization.ExtensionDataObject
AlternateEmailAddresses             :
AlternateMobilePhones               : {}
AlternativeSecurityIds              : {}
BlockCredential                     : False
City                                :
CloudExchangeRecipientDisplayType   : 1073741824
Country                             : US
Department                          :
```

```
DirSyncProvisioningErrors                : {}
DisplayName                              : Damian Scoles
Errors                                   :
Fax                                      :
FirstName                                : Damian
ImmutableId                              :
IndirectLicenseErrors                    : {}
IsBlackberryUser                         : False
IsLicensed                               : True
LastDirSyncTime                          :
LastName                                 : Scoles
LastPasswordChangeTimestamp              : 12/6/2017 6:37:28 AM
LicenseReconciliationNeeded              : False
Licenses                                 : {OnlineExchangeBook:ENTERPRISEPACK}
LiveId                                   : 1003BFFDA6AA7561
MSExchRecipientTypeDetails               :
MobilePhone                              :
ObjectId                                 : ce59bace-e444-4ade-940d-ddbb3754d3c3
Office                                   :
OverallProvisioningStatus                : Success
PasswordNeverExpires                     :
PasswordResetNotRequiredDuringActivate   :
PhoneNumber                              : 630-299-9473
PortalSettings                           : PortalSettings
PostalCode                               :
PreferredDataLocation                    :
PreferredLanguage                        : en
ProxyAddresses                           : {SMTP:damian@OnlineExchangeBook.onmicrosoft.com}
ReleaseTrack                             :
ServiceInformation                       : {}
SignInName                               : damian@OnlineExchangeBook.onmicrosoft.com
SoftDeletionTimestamp                    :
State                                    :
StreetAddress                            :
StrongAuthenticationMethods              : {}
StrongAuthenticationPhoneAppDetails      : {}
StrongAuthenticationProofupTime          :
StrongAuthenticationRequirements         : {}
StrongAuthenticationUserDetails          :
StrongPasswordRequired                   :
StsRefreshTokensValidFrom                : 12/6/2017 6:37:28 AM
Title                                    :
UsageLocation                            : US
UserLandingPageIdentifierForO365Shell    :
UserPrincipalName                        : damian@OnlineExchangeBook.onmicrosoft.com
UserThemeIdentifierForO365Shell          :
UserType                                 : Member
ValidationStatus                         : Healthy
WhenCreated                              : 12/6/2017 6:37:29 AM
```

Get-ExOMailbox

Lists mailbox enabled objects, these can be of RecipientTypeDetail UserMailbox, Shared, Linked, Room or Equipment. The focus of this cmdlet is settings directly related to the mailbox functionality. Most interesting attributes are those related to quotas of not only the user mailbox but all kinds of quotas, mailflow handling, auditing, and the custom attributes.

```
Get-ExOMailbox -Identity Damian@OnlineExchangeBook.OnMicrosoft.Com -PropertySets All | Fl
```

```
ExternalDirectoryObjectId             : 5fc8abf1-2178-4043-b601-4802e0a703cd
Database                              : NAMPR22DG066-db030
MailboxProvisioningConstraint         :
IsMonitoringMailbox                   : False
MailboxRegion                         :
MailboxRegionLastUpdateTime           :
MessageRecallProcessingEnabled        : True
MessageCopyForSentAsEnabled           : False
MessageCopyForSendOnBehalfEnabled     : False
MailboxProvisioningPreferences        : {}
UseDatabaseRetentionDefaults          : False
RetainDeletedItemsUntilBackup         : False
DeliverToMailboxAndForward            : False
IsExcludedFromServingHierarchy        : False
IsHierarchyReady                      : True
IsHierarchySyncEnabled                : True
HasSnackyAppData                      : False
LitigationHoldEnabled                 : False
SingleItemRecoveryEnabled             : True
RetentionHoldEnabled                  : False
EndDateForRetentionHold               :
StartDateForRetentionHold             :
```

```
RetentionComment                    :
RetentionUrl                        :
LitigationHoldDate                  :
LitigationHoldOwner                 :
ElcProcessingDisabled               : False
ComplianceTagHoldApplied            : False
WasInactiveMailbox                  : False
DelayHoldApplied                    : False
InactiveMailboxRetireTime           :
OrphanSoftDeleteTrackingTime        :
LitigationHoldDuration              : Unlimited
ManagedFolderMailboxPolicy          :
RetentionPolicy                     :
AddressBookPolicy                   :
CalendarRepairDisabled              : False
ExchangeGuid                        : 3a5b584b-5f64-4e82-abde-ac32da667965
MailboxContainerGuid                :
UnifiedMailbox                      :
MailboxLocations                    : {1;3a5b584b-5f64-4e82-abde-ac32da667965;Primary;namprd22.prod.outlook.com;53b9
                                      8077-9886-48f6-a2e4-b794d2b89a06, 1;90689132-499c-4d36-b7da-8187251894e6;MainA
                                      rchive;namprd22.prod.outlook.com;3dbb3f7b-b8a5-492f-9093-685c4e39fc96}
AggregatedMailboxGuids              : {}
ExchangeSecurityDescriptor          : System.Security.AccessControl.RawSecurityDescriptor
ExchangeUserAccountControl          : None
MessageTrackingReadStatusEnabled    : True
ExternalOofOptions                  : External
ForwardingAddress                   :
ForwardingSmtpAddress               :
RetainDeletedItemsFor               : 30.00:00:00
IsMailboxEnabled                    : True
Languages                           : {en-US}
OfflineAddressBook                  :
ProhibitSendQuota                   : 99 GB (106,300,440,576 bytes)
ProhibitSendReceiveQuota            : 100 GB (107,374,182,400 bytes)
RecoverableItemsQuota               : 30 GB (32,212,254,720 bytes)
RecoverableItemsWarningQuota        : 20 GB (21,474,836,480 bytes)
CalendarLoggingQuota                : 6 GB (6,442,450,944 bytes)
DowngradeHighPriorityMessagesEnabled : False
ProtocolSettings                    : {POP3§0§§§§§§§§§§§§1, IMAP4§0§§§§§§§§§§§§, ECP§1, MAPI§1§0§§§0§§§§§0...}
RecipientLimits                     : 500
ImListMigrationCompleted            : False
SiloName                            :
IsResource                          : False
IsLinked                            : False
IsShared                            : False
IsRootPublicFolderMailbox           : False
LinkedMasterAccount                 :
ResetPasswordOnNextLogon            : False
ResourceCapacity                    :
ResourceCustom                      : {}
ResourceType                        :
RoomMailboxAccountEnabled           :
SamAccountName                      : Damia576921572707534
SCLDeleteThreshold                  :
SCLDeleteEnabled                    :
SCLRejectThreshold                  :
SCLRejectEnabled                    :
SCLQuarantineThreshold              :
SCLQuarantineEnabled                :
SCLJunkThreshold                    :
SCLJunkEnabled                      :
AntispamBypassEnabled               : False
ServerLegacyDN                      : /o=ExchangeLabs/ou=Exchange Administrative Group
                                      (FYDIBOHF23SPDLT)/cn=Configuration/cn=Servers/cn=CY4PR22MB0407
UseDatabaseQuotaDefaults            : False
IssueWarningQuota                   : 98 GB (105,226,698,752 bytes)
RulesQuota                          : 256 KB (262,144 bytes)
Office                              : [redacted]
UserPrincipalName                   : damian@[redacted]
UMEnabled                           : False
```

```
MaxSafeSenders                      :
MaxBlockedSenders                   :
NetID                               : 10033FFF86C787F0
ReconciliationId                    :
WindowsLiveID                       : damian@
MicrosoftOnlineServicesID           : damian@
ThrottlingPolicy                    :
RoleAssignmentPolicy                : Default Role Assignment Policy
DefaultPublicFolderMailbox          :
EffectivePublicFolderMailbox        : PF-O365-01
SharingPolicy                       : Default Sharing Policy
RemoteAccountPolicy                 :
MailboxPlan                         : ExchangeOnlineEnterprise-a198b3f1-d047-49ed-b00d-b9ae84d0390a
ArchiveDatabase                     : NAMPR22DG133-db084
ArchiveGuid                         : 90689132-499c-4d36-b7da-8187251894e6
ArchiveName                         : {OnPremArchive-Damian}
JournalArchiveAddress               :
ArchiveQuota                        : 100 GB (107,374,182,400 bytes)
ArchiveWarningQuota                 : 90 GB (96,636,764,160 bytes)
ArchiveDomain                       :
ArchiveStatus                       : None
ArchiveState                        : Local
AutoExpandingArchiveEnabled         : False
DisabledMailboxLocations            : False
RemoteRecipientType                 : Migrated
DisabledArchiveDatabase             :
DisabledArchiveGuid                 : 00000000-0000-0000-0000-000000000000
QueryBaseDN                         :
QueryBaseDNRestrictionEnabled       : False
MailboxMoveTargetMDB                :
MailboxMoveSourceMDB                :
MailboxMoveFlags                    : None
MailboxMoveRemoteHostName           :
MailboxMoveBatchName                :
MailboxMoveStatus                   : None
MailboxRelease                      : E15
ArchiveRelease                      : E15
IsPersonToPersonTextMessagingEnabled : False
```

Get-MailboxAutoReplyConfiguration

Configuration of the Out of Office (OOF) replies, including scheduling, and inside and outside organization message. Basically, every possible setting the user (or an admin) has set. With this you can check whether an OOF has been set.

```
Get-MailboxAutoReplyConfiguration -Identity Gene.Ricks@Contoso.Com
```

```
RunspaceId                      : 223be553-ccd2-4a1c-9a00-4427470d894d
AutoDeclineFutureRequestsWhenOOF : False
AutoReplyState                  : Disabled
CreateOOFEvent                  : False
DeclineAllEventsForScheduledOOF : False
DeclineEventsForScheduledOOF    : False
EventsToDeleteIDs               :
EndTime                         : 12/8/2017 12:00:00 AM
ExternalAudience                : All
ExternalMessage                 :
InternalMessage                 :
DeclineMeetingMessage           :
OOFEventSubject                 :
StartTime                       : 12/7/2017 12:00:00 AM
MailboxOwnerId                  : Gene Ricks
Identity                        : Gene Ricks
IsValid                         : True
ObjectState                     : Unchanged
```

Get-CalendarProcessing

Display the way Exchange will process meeting invites on the mailbox at hand. In most cases for user mailboxes the default settings will be adequate, however for Room, Equipment and maybe Shared mailboxes changes may be required. Due to privacy regulations, it might be required to remove the subject of a meeting from a room mailbox. Or you want to limit the way users are booking a meeting with a room mailbox.

```
Get-CalendarProcessing -Identity Gene.Ricks@Contoso.Com | Fl
```

```
RunspaceId                          : 223be553-ccd2-4a1c-9a00-4427470d894d
AutomateProcessing                  : AutoUpdate
AllowConflicts                      : False
BookingWindowInDays                 : 180
MaximumDurationInMinutes            : 1440
AllowRecurringMeetings              : True
EnforceSchedulingHorizon            : True
ScheduleOnlyDuringWorkHours         : False
ConflictPercentageAllowed           : 0
MaximumConflictInstances            : 0
ForwardRequestsToDelegates          : True
DeleteAttachments                   : True
DeleteComments                      : True
RemovePrivateProperty               : True
DeleteSubject                       : True
AddOrganizerToSubject               : True
DeleteNonCalendarItems              : True
TentativePendingApproval            : True
EnableResponseDetails               : True
OrganizerInfo                       : True
ResourceDelegates                   : {}
RequestOutOfPolicy                  : {}
AllRequestOutOfPolicy               : False
BookInPolicy                        : {}
AllBookInPolicy                     : True
RequestInPolicy                     : {}
AllRequestInPolicy                  : False
AddAdditionalResponse               : False
AdditionalResponse                  :
RemoveOldMeetingMessages            : True
AddNewRequestsTentatively           : True
ProcessExternalMeetingMessages      : False
RemoveForwardedMeetingNotifications : False
MailboxOwnerId                      : Gene Ricks
Identity                            : Gene Ricks
```

Get-MailboxCalendarConfiguration

This cmdlet shows the configuration of the calendar specific settings, such as the time zone, working hours and such. You could use this to check whether users are correctly provisioned per their actual regional location or other considerations.
Most of these features influence Outlook Web App, although some are also valid for other clients. The EventsFromEmailEnabled* and Weather* settings are for Exchange Online only (as some other features which are mention on the Microsoft Docs page for this cmdlet).

Note that this cmdlet does not change any calendar processing settings. See Get-CalendarProcessing for those settings.

```
Get-MailboxCalendarConfiguration -Identity Gene.Ricks@Contoso.Com | Fl
```

```
RunspaceId                  : 223be553-ccd2-4a1c-9a00-4427470d894d
WorkDays                    : Weekdays
WorkingHoursStartTime       : 08:00:00
WorkingHoursEndTime         : 17:00:00
WorkingHoursTimeZone        : Pacific Standard Time
WeekStartDay                : Monday
ShowWeekNumbers             : False
FirstWeekOfYear             : FirstDay
TimeIncrement               : ThirtyMinutes
RemindersEnabled            : True
ReminderSoundEnabled        : True
DefaultReminderTime         : 00:15:00
WeatherEnabled              : FirstRun
```

```
WeatherUnit                           : Default
WeatherLocations                      : {}
WeatherLocationBookmark               : 0
DefaultMeetingDuration                : 30
AgendaMailEnabled                     : False
SkipAgendaMailOnFreeDays              : True
DailyAgendaMailSchedule               : Default
AgendaMailIntroductionEnabled         : True
EventsFromEmailEnabled                : True
EventsFromEmailDelegateChecked        : False
EventsFromEmailShadowMailboxChecked   : False
ReportEventsCreatedFromEmailEnabled   : True
CreateEventsFromEmailAsPrivate        : True
FlightEventsFromEmailEnabled          : True
DiningEventsFromEmailEnabled          : True
HotelEventsFromEmailEnabled           : True
RentalCarEventsFromEmailEnabled       : True
EntertainmentEventsFromEmailEnabled   : True
PackageDeliveryEventsFromEmailEnabled : False
InvoiceEventsFromEmailEnabled         : True
UseBrightCalendarColorThemeInOwa      : False
CalendarFeedsPreferredLanguage        :
CalendarFeedsPreferredRegion          :
CalendarFeedsRootPageId               :
ConversationalSchedulingEnabled       : True
IsMailboxSectionEnabled               : True
IsCalendarSectionEnabled              : True
IsWorkingHoursSectionEnabled          : True
LocalEventsEnabled                    : FirstRun
LocalEventsLocation                   :
AgendaPaneEnabled                     : True
Identity                              : Gene Ricks
```

Get-MailboxCalendarFolder

The cmdlet shows settings specifically targeted at sharing or publishing Calendar folder data of the user. It shows the period that data is visible, including the detail level for anonymous users. This is only the case when the calendar is shared, which is defined by the PublishEnabled attribute and the existence of publishing URLs.

```
Get-MailboxCalendarFolder -Identity Gene.Ricks@Contoso.Com:\Calendar
```

```
Identity                        : Gene Ricks:\Calendar
PublishEnabled                  : False
PublishDateRangeFrom            : ThreeMonths
PublishDateRangeTo              : ThreeMonths
DetailLevel                     : LimitedDetails
SearchableUrlEnabled            : False
PublishedCalendarUrl            :
PublishedICalUrl                :
ExtendedFolderFlags             : ExchangePublishedCalendar
CalendarSharingFolderFlags      : None
CalendarSharingOwnerSmtpAddress :
CalendarSharingPermissionLevel  : Null
SharingLevelOfDetails           : None
SharingPermissionFlags          : None
SharingOwnerRemoteFolderId      : AAA=
LastAttemptedSyncTime           : 01/02/0001 00:00:00
LastSuccessfulSyncTime          : 01/02/0001 00:00:00
```

The same user now with a shared calendar:

```
Identity                        : Gene Ricks:\Calendar
PublishEnabled                  : True
PublishDateRangeFrom            : ThreeMonths
PublishDateRangeTo              : ThreeMonths
DetailLevel                     : LimitedDetails
SearchableUrlEnabled            : False
PublishedCalendarUrl            : http://outlook.office365.com/owa/calendar/79fbc71d49a4487fb21bffc
                                  7e5dd6072@OnlineExchangeBook.onmicrosoft.com/3ed26c2e01194f5caa4b
                                  6d5b89ffe0de10347291611370809252/calendar.html
PublishedICalUrl                : http://outlook.office365.com/owa/calendar/79fbc71d49a4487fb21bffc
                                  7e5dd6072@OnlineExchangeBook.onmicrosoft.com/3ed26c2e01194f5caa4b
                                  6d5b89ffe0de10347291611370809252/calendar.ics
ExtendedFolderFlags             : ExchangePublishedCalendar
CalendarSharingFolderFlags      : None
CalendarSharingOwnerSmtpAddress :
CalendarSharingPermissionLevel  : Null
SharingLevelOfDetails           : None
SharingPermissionFlags          : None
SharingOwnerRemoteFolderId      : AAA=
LastAttemptedSyncTime           : 01/02/0001 00:00:00
LastSuccessfulSyncTime          : 01/02/0001 00:00:00
```

Get-MailboxFolder
View information on folders in your own mailbox.

Get-MailboxFolder Damian:\Inbox | Fl

```
RunspaceId                  : 9ad6a38d-fd12-42e0-9bc4-9b9e8114c0d9
Name                        : Inbox
Identity                    : damian:\Inbox
ParentFolder                : damian:\
FolderStoreObjectId         : LgAAAAAHDC2POPk+RbUOQxp8hiH0AQC3MDdN6NBeTbIx9ZB1/izjAAAAAAEMAAAB
FolderSize                  : 18501
HasSubfolders               : False
FolderClass                 : IPF.Note
FolderPath                  : {Inbox}
AssociatedDumpsterFolders   :
DefaultFolderType           : Inbox
ExtendedFolderFlags         : Normal
MailboxOwnerId              : damian
IsValid                     : True
ObjectState                 : Unchanged
```

Do note that you require the correct permissions on the mailbox, otherwise an error will be shown stating that the mailbox doesn't exist. It's already trying to get information on the root folder and because you don't have access it will think it doesn't exist. The default permissions are set via the Role Based Access Control role MyBaseOptions, this means that even an administrator can only use this cmdlet on their own mailbox, but will get this error when trying to query others. This obviously limits the use of this cmdlet for reporting.

```
The specified mailbox  Gene.Ricks@Contoso.Com\Inbox doesn't exist.
    + CategoryInfo          : NotSpecified: (:) [Get-MailboxFolder], ManagementObjectNotFoundExcep
   tion
    + FullyQualifiedErrorId : [Server=L16-EX01,RequestId=93e85a5d-0e0e-4210-b8dd-16577b1fd465,T
   imeStamp=12/17/2016 3:04:26 PM] [FailureCategory=Cmdlet-ManagementObjectNotFoundException] 701B
  F26B,Microsoft.Exchange.Management.StoreTasks.GetMailboxFolder
    + PSComputerName        : L16-EX01.LAb2016.Com
```

Get-ExOMailboxFolderPermission
Used to view folder permissions within mailboxes. You must specify the correct folder path, which for the default well-known folders in Exchange is dependent on the regional settings of the mailbox that create these folders at first login. So, the default Calendar folder might be named different in Spanish.

Also, note that the Calendar folder permissions have additional AccessRights available, AvailabilityOnly and LimitedDetails. Both influence the visibility of specific information of meetings (subject and location is also shown with LimitedDetails).

```
Get-MailboxFolderPermission -Identity Gene.Ricks@Contoso.Com:\Inbox
```

```
PS C:\> Get-MailboxFolderPermission -Identity Gene.Ricks@Contoso.Com:\Inbox |ft

Identity          FolderName User    AccessRights SharingPermissionFlags
--------          ---------- ----    ------------ ----------------------
Gene.Ricks:\Inbox Inbox      Default {None}

PS C:\> Get-MailboxFolderPermission -Identity Gene.Ricks@Contoso.Com:\Calendar |ft

Identity             FolderName User             AccessRights       SharingPermissionFlags
--------             ---------- ----             ------------       ----------------------
Gene.Ricks:\Calendar Calendar   Default          {AvailabilityOnly}
Gene.Ricks:\Calendar Calendar   TUser@Contoso.Com{Editor}
```

Get-ExOMailboxFolderStatistics

View information on specific folders in a mailbox. This includes the folder size and number of items. For more information, you can add the -IncludeAnalysis switch, which can help with troubleshooting. It will return values that would otherwise remain empty, the reason being that it can take a while for the analysis to complete. The values, however, can help with troubleshooting or reporting. (This example is from the legacy cmdlet as it is still accessible and is the only version that currently allows 'IncludeAnalysis').

```
Get-MailboxFolderStatistics -Identity Gene.Ricks@Contoso.Com -FolderScope Inbox
 -IncludeAnalysis
```

```
TopSubject                      : [Outlook junk mail report] isthis sapm is this spam?
TopSubjectSize                  : 30.29 KB (31,015 bytes)
TopSubjectCount                 : 1
TopSubjectClass                 : REPORT.IPM.Note.NDR
TopSubjectPath                  : \Top of Information Store\Inbox
TopSubjectReceivedTime          : 12/9/2016 1:02:52 AM
TopSubjectFrom                  : Microsoft Outlook
TopClientInfoForSubject         :  \  \
TopClientInfoCountForSubject    : 1
```

Another parameter that might provide useful information for troubleshooting or reporting is the IncludeOldestAndNewestItems parameter. As the name suggests, you will then receive more information on the oldest and newest items in the specified mailbox.

```
Get-ExOMailboxFolderStatistics -Identity Gene.Ricks@Contoso.Com -FolderScope Inbox
 -IncludeOldestAndNewestItems
```

```
Name                            : Inbox
FolderPath                      : /Inbox
FolderId                        : LgAAAAAwdsyynLpiQrW+WjOEe0DOAQC0KYIxIq2XTo+/o8YVVNf+AAAAAAEMAAAB
ParentFolderId                  : LgAAAAAwdsyynLpiQrW+WjOEe0DOAQC0KYIxIq2XTo+/o8YVVNf+AAAAAAEIAAAB
FolderType                      : Inbox
ContentFolder                   : True
ContentMailboxGuid              : 1c668041-f06c-4562-bc68-51d07b8c50ba
RawContentMailboxGuid           :
Movable                         : False
RecoverableItemsFolder          : False
AssociatedIPMFolderPath         :
ContainerClass                  :
Flags                           :
TargetQuota                     : User
StorageQuota                    : Unlimited
StorageWarningQuota             : Unlimited
VisibleItemsInFolder            : 986
HiddenItemsInFolder             : 46
ItemsInFolder                   : 1032
DeletedItemsInFolder            : 0
FolderSize                      : 117.7 MB (123,451,985 bytes)
ItemsInFolderAndSubfolders      : 1108
DeletedItemsInFolderAndSubfolders : 0
FolderAndSubfolderSize          : 130.1 MB (136,459,835 bytes)
CurrentSchemaVersion            : 1.50
OldestItemReceivedDate          : 8/4/2017 7:10:19 PM
NewestItemReceivedDate          : 11/12/2020 4:40:20 AM
OldestDeletedItemReceivedDate   :
NewestDeletedItemReceivedDate   :
OldestItemLastModifiedDate      : 10/12/2018 3:51:31 AM
NewestItemLastModifiedDate      : 11/12/2020 4:40:21 AM
```

Note: You do not supply a folder path, but rather a folder type with the FolderScope parameter. With this all folders of the same type are returned and not just one specific folder.

Valid input values for FolderScope are:

```
All                 Contacts                DeletedItems            Inbox
Journal             ManagedCustomFolder     Notes                   Personal
RssSubscriptions    SyncIssues              Calendar                ConversationHistory
Drafts              JunkEmail               LegacyArchiveJournals   NonIpmRoot
Outlook             RecoverableItems        SentItems               Tasks
```

The ManagedCustomFolder value returns output for all managed custom folders. The RecoverableItems value returns output for the Recoverable Items folder and the Deletions, DiscoveryHolds, Purges, and Versions subfolders. Also see Microsoft Docs. If you require information regarding statistics of the whole mailbox, see Get-ExOMailboxStatistics.

Remove-CalendarEvents
"This cmdlet cancels meetings in the specified mailbox where the mailbox is the meeting organizer, and the meeting has one or more attendees or resources. It doesn't cancel appointments or meetings without attendees or resources. Because meeting cancellations must be sent out, the mailbox must still be enabled to send mail."

Examples for this cmdlet:

```
-------------------------- Example 1 --------------------------

Remove-CalendarEvents -Identity chris@contoso.com -CancelOrganizedMeetings

-------------------------- Example 2 --------------------------

Remove-CalendarEvents -Identity "Angela Gruber" -CancelOrganizedMeetings -QueryStartDate 11-1-2018
-QueryWindowInDays 120
-------------------------- Example 3 --------------------------

Remove-CalendarEvents -Identity "Jacob Berger" -CancelOrganizedMeetings -QueryStartDate 9-1-2018
-QueryWindowInDays 90 -PreviewOnly -Verbose
```

From the above examples, we see that there are a few parameters we can choose from to run this cmdlet:

CancelOrganizedMeetings: use this to cancel meetings in the selected mailbox
PreviewOnly: Allows a preview of the potential changes without making them
QueryStartDate: Specify a starting date to look for meetings to remove
QueryWindowInDays: Number of days to look for meetings after the start date from the QueryStartDate

Let's take a scenario where we need to remove all meetings that were organized by an End User, we know they are retiring or quitting on April 1, 2021. We can first preview these changes like so:

```
Remove-CalendarEvents -Identity Administrator -PreviewOnly
```

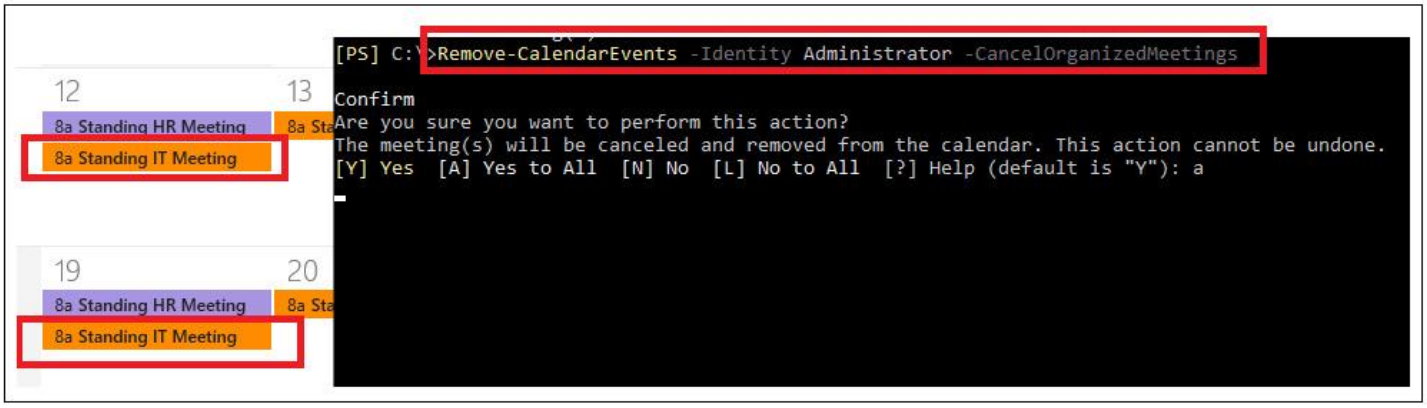
```
[PS] C:\>Remove-CalendarEvents -Identity Administrator -CancelOrganizedMeetings -PreviewOnly

Confirm
Are you sure you want to perform this action?
The meeting(s) will be canceled and removed from the calendar. This action cannot be undone.
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [?] Help (default is "Y"): y
The recurring meeting with subject "Standing IT Meeting" has been queued for cancellation.
```

```
Remove-CalendarEvents -Identity Administrator -CancelOrganizedMeetings -WhatIf
```

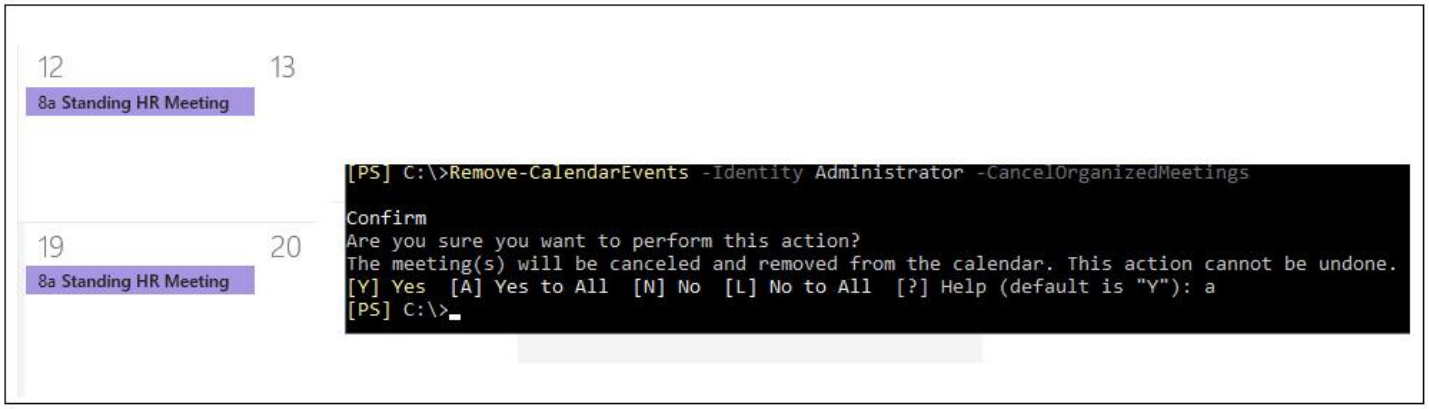
```
[PS] C:\>Remove-CalendarEvents -Identity Administrator -CancelOrganizedMeetings -WhatIf
What if: The meeting(s) will be canceled and removed from the calendar. This action cannot be undone.
[PS] C:\>
```

Before and after removal:

Before deletion



After deletion



Get-MailboxJunkEmailConfiguration

Use this cmdlet to see the User Junk Mail folder configuration for a specific mailbox, including any blocked or trusted email addresses or domains. This can be useful to determine whether your central anti-spam solutions require some tweaking.

```
Get-MailboxJunkEmailConfiguration -Identity Gene.Ricks@Contoso.Com
```

In this case the user has blocked a Wingtoys address.

```
RunspaceId                    : 9ad6a38d-fd12-42e0-9bc4-9b9e8114c0d9
Status                        : IsPresent, IsEnabled
Enabled                       : True
TrustedListsOnly              : False
ContactsTrusted               : False
TrustedSendersAndDomains      : {fabrikam.com}
BlockedSendersAndDomains      : {henk@wingtoys.com}
TrustedRecipientsAndDomains   : {fabrikam.com}
MailboxOwnerId                : Gene Ricks
Identity                      : Gene Ricks
IsValid                       : True
ObjectState                   : Unchanged
```

Note: Realize that with Exchange Online Protection, these settings can require an additional step in troubleshooting messages ending up in the Junk Mail folder. See this Microsoft article: https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/anti-spam-protection

Get-MailboxMessageConfiguration

Shows the configuration of Outlook Web App for a specific mailbox.

```
Get-MailboxMessageConfiguration -Identity Gene.Ricks@Contoso.Com
```

```
RunspaceId                      : 9ad6a38d-fd12-42e0-9bc4-9b9e8114c0d9
AfterMoveOrDeleteBehavior       : OpenNextItem
NewItemNotification             : All
EmptyDeletedItemsOnLogoff       : False
AutoAddSignature                : True
AutoAddSignatureOnReply         : False
SignatureText                   :
SignatureHtml                   :
AutoAddSignatureOnMobile        : True
SignatureTextOnMobile           :
UseDefaultSignatureOnMobile     : True
DefaultFontName                 : Calibri
DefaultFontSize                 : 3
DefaultFontColor                : #000000
DefaultFontFlags                : Normal
AlwaysShowBcc                   : False
AlwaysShowFrom                  : True
DefaultFormat                   : Html
ReadReceiptResponse             : DoNotAutomaticallySend
PreviewMarkAsReadBehavior       : OnSelectionChange
PreviewMarkAsReadDelaytime      : 5
ConversationSortOrder           : ChronologicalNewestOnTop
ShowConversationAsTree          : False
HideDeletedItems                : False
SendAddressDefault              :
EmailComposeMode                : Inline
CheckForForgottenAttachments    : True
AreFlaggedItemsPinned           : False
IsReplyAllTheDefaultResponse    : False
KeyboardShortcutsMode           : Owa
LinkPreviewEnabled              : True
ShowPreviewTextInListView       : True
ShowUpNext                      : True
GlobalReadingPanePosition       : Right
IsFavoritesFolderTreeCollapsed  : False
IsMailRootFolderTreeCollapsed   : False
MailFolderPaneExpanded          : True
IsHashtagTreeCollapsed          : False
IsGroupsTreeCollapsed           : False
GroupSuggestionDismissalCount   : 0
GroupSuggestionDismissalDate    :
ShowSenderOnTopInListView       : True
ShowReadingPaneOnFirstLoad      : False
NavigationPaneViewOption        : Default
AllOwaConfiguration             :
PreferAccessibleContent         : False
MailboxOwnerId                  : Gene Ricks
Identity                        : Gene Ricks
IsValid                         : True
ObjectState                     : Unchanged
```

Get-ExOMailboxPermission

Shows the permissions set on the specific mailbox. Note that these are not permissions on the subsequent folders. The IsInherited column indicates whether the permission is inherited from a higher source from the Azure Active Directory (AAD) configuration as Mailbox permissions are actually AAD permissions.

```
Get-ExOMailboxPermission -Identity Gene.Ricks@Contoso.Com
```

```
PS C:\> Get-MailboxPermission -Identity Gene.Ricks@contoso.com | Where {$_.IsInherited -eq $False}

Identity            User                AccessRights                IsInherited Deny
--------            ----                ------------                ----------- ----
Gene Ricks          NT AUTHORITY\SELF   {FullAccess, ReadPermission} False       False
```

If you require only the permissions of a specific user, you can use the -User parameter.

```
Get-ExOMailboxPermission -Identity Gene.Ricks@Contoso.Com -User Damian
```

```
Identity     User                    AccessRights                IsInherited Deny
--------     ----                    ------------                ----------- ----
Gene Ricks   S-1-5-21-1824196590-355 {FullAccess}                      False True
Gene Ricks   NT AUTHORITY\SELF       {FullAccess, ReadPermission}      False False
Gene Ricks   damian@contoso.com      {FullAccess}                      False False
```

In some cases, you only want to report on non-inherited permissions i.e. directly assigned mailbox permissions, which are the permissions set if you use Exchange cmdlets. You can do that by filtering using the Where cmdlet.

```
Get-ExOMailboxPermission -Identity Gene.Ricks@Contoso.Com | Where {$_.IsInherited
 -eq $False}
```

Get-MailboxRegionalConfiguration

Use this cmdlet to extract regional settings on a specific mailbox, such as time zone, date format, language etc..

```
Get-MailboxRegionalConfiguration -Identity Gene.Ricks@Contoso.Com | fl
```

```
DateFormat                                : M/d/yyyy
Language                                  : en-US
DefaultFolderNameMatchingUserLanguage     : False
TimeFormat                                : h:mm tt
TimeZone                                  : W. Europe Standard Time
Identity                                  : Gene Ricks
```

The *DefaultFolderNameMatchingUserLanguage* indicates whether the default (or Well-Known folders such as Inbox) are localized, if True, those folders names are in the language indicated. This has an impact when you use specific cmdlets that target specific folders, for instance folder permissions.

Get-MailboxSpellingConfiguration

Retrieve spelling configuration set by the user for Outlook Web App.

```
Get-MailboxSpellingConfiguration -Identity Gene.Ricks@Contoso.Com
```

```
RunspaceId          : c8ca4800-9430-4aea-aeab-a740cc956b4c
CheckBeforeSend     : False
DictionaryLanguage  : EnglishUnitedStates
IgnoreUppercase     : False
IgnoreMixedDigits   : False
Identity            : Gene Ricks
IsValid             : True
ObjectState         : New
```

Get-ExOMailboxStatistics

Will show you statistics of a specific mailbox, such as the Display Name, DeletedItemCount, TotalItemSize etc. (although if this interests you, you should turn on auditing on those mailboxes for more detail information).

```
Get-EXOMailboxStatistics damian@practicalpowershell.com
```

```
DisplayName          : Damian-PP
MailboxGuid          : 1c668041-f06c-4562-bc68-51d07b8c50ba
DeletedItemCount     : 54
ItemCount            : 1889
TotalDeletedItemSize : 2.678 MB (2,807,675 bytes)
TotalItemSize        : 139.4 MB (146,223,881 bytes)
```

Retrieve client access settings on a specific mailbox such as what kind of protocols are enabled on this mailbox and the specific configuration of these protocols:

```
Get-ExOCASMailbox -Identity Gene.Ricks@Contoso.Com Ft Name, ActiveSyncE*, OWA*, PopE*,
 ImapE*, MAPIE* -Auto
```

```
Name        ActiveSyncEnabled OWAEnabled PopEnabled ImapEnabled MapiEnabled
----        ----------------- ---------- ---------- ----------- -----------
Gene Ricks  False             True       False      False       True
```

```
Get-ExOCASMailbox -Identity damian@practicalpowershell.com -PropertySets All | Fl
```

```
UniversalOutlookEnabled                 : True
OutlookMobileEnabled                    : True
MacOutlookEnabled                       : True
ECPEnabled                              : True
OWAforDevicesEnabled                    : True
ShowGalAsDefaultView                    : True
EmailAddresses                          : {X500:/o=First Organization/ou=External (FYDI
                                          (FYDIBOHF23SPDLT)/cn=Recipients/cn=c8b32fe3d9
LegacyExchangeDN                        : /o=ExchangeLabs/ou=Exchange Administrative Gr
LinkedMasterAccount                     :
SamAccountName                          : Damia57516-363895667
SmtpClientAuthenticationDisabled        :
OWAEnabled                              : True
DistinguishedName                       : CN=Damian,OU=scoles.onmicrosoft.com,OU=Mic
ExchangeObjectId                        : 343b634d-1261-4700-8b68-9f5e062c785d
ObjectCategory                          : NAMPR08A001.prod.outlook.com/Configuration/Sc
ObjectClass                             : {top, person, organizationalPerson, user}
OrganizationId                          : NAMPR08A001.prod.outlook.com/Microsoft Exchan
PublicFolderClientAccess                : False
WhenChangedUTC                          : 11/10/2020 9:22:10 PM
WhenCreatedUTC                          : 3/25/2017 5:53:12 PM
WhenChanged                             : 11/10/2020 3:22:10 PM
WhenCreated                             : 3/25/2017 12:53:12 PM
OwaMailboxPolicy                        : OwaMailboxPolicy-Default
IsOptimizedForAccessibility             :
ImapEnabled                             : False
ImapSuppressReadReceipt                 : False
ImapEnableExactRFC822Size               : False
ImapMessagesRetrievalMimeFormat         : BestBodyFormat
ImapUseProtocolDefaults                 : True
ImapForceICalForCalendarRetrievalOption : False
PopEnabled                              : False
PopSuppressReadReceipt                  : False
PopEnableExactRFC822Size                : False
PopMessagesRetrievalMimeFormat          : BestBodyFormat
```

Test-MAPIConnectivity

This cmdlet is used to test connectivity for a mailbox as well as "the MAPI server, Exchange store, and Directory Service Access (DSAccess) are working". For Exchange on-premises this would be good for testing your own Exchange Servers and when it comes to Exchange Online you are testing Microsoft's own configuration and if there are any issues. For Exchange Online there are switches that simply won't work as they are meant for on-premises environments, so that is something to keep in mind. Now, on to the PowerShell:

```
Get-Help Test-MAPIConnectivity -Examples
```

```
-------------------------- Example 1 --------------------------

Test-MapiConnectivity -Server "Server01"

-------------------------- Example 2 --------------------------

Test-MapiConnectivity -Identity "midwest\john"
```

Even the examples are more geared to on-premises. Well, what parameters do we have for Exchange Online:

Identity: Specify the UPN of a user to test.
Archive: Test the MAPI connection to a user's archive mailbox.

Let's run this against a mailbox in Exchange Online to see what we get:

```
Test-MAPIConnectivity -Identity damian@practicalpowershell.com
```

```
RunspaceId           : ebace8ac-a18d-4d9b-8afd-0ae9d9d21be6
Server               : CH2PR22MB1798
Database             : NAMPR22DG113-db148
Mailbox              : Damian-PP
MailboxGuid          : 1c668041-f06c-4562-bc68-51d07b8c50ba
IsArchive            : False
IsDatabaseCopyActive : True
IsUserLoginAllowed   : True
Result               : Success
Latency              : 00:00:00.2907685
Error                :
```

As we can see from the red rectangle, the test was successful and the latency for the test was extremely low. Now, what if we were to test the user's archive mailbox as well:

```
Test-MAPIConnectivity -Identity damian@practicalpowershell.com -Archive
```

```
RunspaceId           : ebace8ac-a18d-4d9b-8afd-0ae9d9d21be6
Server               : DM5PR2201MB1577
Database             : NAMPR22DG099-db104
Mailbox              : Damian-PP
MailboxGuid          : e979f3bb-8a36-4ebe-a3d8-43e14a59e87f
IsArchive            : True
IsDatabaseCopyActive : True
IsUserLoginAllowed   : True
Result               : Success
Latency              : 00:00:00.1300996
Error                :
Identity             :
```

We see the same success and latency as well as an indicator that the Archive Mailbox was tested in this cmdlet. Otherwise we know both mailbox and archive mailbox are MAPI accessible with these tests.

Recoverable Items

Get-RecoverableItems and Restore-RecoverableItems

A pair of interesting cmdlets was released from Microsoft since the First Edition of this book. These cmdlets specifically deal with emails that are recoverable in a user's mailbox. Typically these emails are found in either the 'Deleted Items' Folder or in 'Recoverable Items\Purges'.

Note: Mailbox Import Export role group should be signed before executing these cmdlets.

First, let's see what the Get-RecoverableItems cmdlet can reveal on a mailbox. As always, we can check either the Get-Help for this cmdlet or we can check the Online help for examples. For this cmdlet, no Get-Help examples exist, so we will explore the Online help located here:

https://docs.microsoft.com/en-us/powershell/module/exchange/mailboxes/get-recoverableitems

This page provides us with two examples to use:

Example 1

```
Get-RecoverableItems -Identity laura@contoso.com -Subject -SubjectContains "FY17
 Accounting" -FilterItemType IPM.Note -FilterStartTime "2/1/2018 12:00:00 AM"
 -FilterEndTime "2/5/2018 11:59:59 PM"
```

Example 2

```
Get-RecoverableItems -Identity "malik@contoso.com", "lillian@contoso.com" -FilterItemType
 IPM.Note -FilterStartTime "3/15/2019 12:00:00 AM" -FilterEndTime "3/25/2019 11:59:59 PM"
```

As we can see from the examples, we can query for recoverable deleted items based on mailbox, item type start and end time and subject.

Item to Filter by:

https://docs.microsoft.com/en-us/office/vba/outlook/concepts/forms/item-types-and-message-classes

Most of the items that are typically deleted are emails, which is listed as 'IPM.Note'. Let's review a couple of mailboxes to see what we can recover and then we will explore the recover cmdlet. First, a quick pull of all items that can be recovered (totals):

```
PS C:\> (Get-RecoverableItems -Identity damian).Count
1558
```

When the cmdlet is running, we briefly see this:

```
Finding recoverable Items… do not interrupt.
   Mailboxes: 0/1      Folders: 0/3      Items found: 0
   [
```

Now, let's explore some properties and types of items that we can find with a simple search first and then we can refine that search as we get more information about these items. If we run the default cmdlet like so:

```
Get-RecoverableItems -Identity Damian
```

This however will provide the data in a Format-List format, which may make it hard to read. Example recoverable message:

```
RunspaceId           : 5a40fa2e-d2d2-4a66-ac91-8aa5d0dca05c
LastParentPath       : Inbox
LastParentFolderID   :
OriginalFolderExists : False
Identity             : 5d0cc54e-0082-4eb8-a300-ce17a036f3f4\3a5b584b-5f64-4e82-abde-ac32da667965
MailboxIdentity      : 5d0cc54e-0082-4eb8-a300-ce17a036f3f4\3a5b584b-5f64-4e82-abde-ac32da667965
ItemClass            : IPM.Schedule.Meeting.Resp.Neg.SilentResponse
Subject              : Declined: Azure Cognitive Search - ML Enabled Ranking - Option 2
EntryID              : 00000000685800CDD60E98469824E2CB31775EAA0700B0EE11A4CF02344BBB700CAF396B678B0000000001B3
SourceFolder         : Recoverable Items\Purges
LastModifiedTime     : 12/31/2019 05:37:51
```

Notice that there are some vital clues and properties that can assist us later. These properties are 'LastParentPath', 'subject', 'ItemClass' and 'SourceFolder'. We can use these fields to help us focus on a particular item, item type or whatever. One thing to note is that outputting this to a Format Table format may not be the most useful way without choosing fields. This is because the first default fields to display are RunspaceId, LastParentPath, LastParentFolderID, OriginalFolderExists and Identity.

We can also quickly explore the range of dates available to recover from a mailbox by gathering the first and last date for the Recoverable Items. The most recent event is Event 0 and the very last Event is [-1].

Note: The [-1] value of an array is the last item in the array whereas [0] is the very first item in the array.

```
$AllRecoverable = Get-RecoverableItems -Identity damian
$Dates = $AllRecoverable.LastModifiedTime
$Last = $Dates[0]
$First = $Dates[-1]
```

```
PS C:\> $AllRecoverable = Get-RecoverableItems -Identity damian
PS C:\> $Dates = $AllRecoverable.LastModifiedTime
PS C:\> $Dates[0]
01/08/2020 19:00:26
PS C:\> $Dates[-1]
12/31/2019 05:37:51
```

We see from the above, the farther back we can look at for Recoverable Items in this mailbox is 12/31/2019 @ 5:37 AM. Whereas the most recent item that is recoverable is from 1/8/2020 @ 7:00 PM.

Now, we can get counts of message types if we want to as well:

```
(Get-RecoverableItems -Identity damian -FilterItemType 'IPM.Note').Count
(Get-RecoverableItems -Identity damian -FilterItemType 'IPM.Appointment').Count
(Get-RecoverableItems -Identity damian -FilterItemType
 'IPM.Schedule.Meeting.Resp.Neg.SilentResponse').Count
(Get-RecoverableItems -Identity damian -FilterItemType 'IPM.SharePointItem').Count
```

Which provides these results:

```
PS C:\> (Get-RecoverableItems -Identity damian -FilterItemType 'IPM.Note').Count
1531
PS C:\> (Get-RecoverableItems -Identity damian -FilterItemType 'IPM.Appointment').Count
6
PS C:\> (Get-RecoverableItems -Identity damian -FilterItemType 'IPM.Schedule.Meeting.Resp.Neg.SilentResponse').Count
5
PS C:\> (Get-RecoverableItems -Identity damian -FilterItemType 'IPM.SharePointItem').Count
4
```

We can see that, at least for this mailbox, the IPM.Note Item Type is the majority of the items to filter for. Now that we see how we can use the Get-RecoverableItems cmdlet, let's explore the Restore-RecoverableItems cmdlet to see what we can do with these items.

Restore-RecoverableItems

First, let's review any examples we can find. Like the previous cmdlet, there is no examples from Get-Help. However, we can find examples from the Microsoft Docs page for the cmdlet, which is located here:

https://docs.microsoft.com/en-us/powershell/module/exchange/mailboxes/restore-recoverableitems

Example 1

```
Restore-RecoverableItems -Identity laura@contoso.com -FilterItemType IPM.Note
 -SubjectContains "FY18 Accounting" -FilterStartTime "2/1/2018 12:00:00 AM" -FilterEndTime
 "2/5/2018 11:59:59 PM"
```

Example 2

```
$Mailboxes = Import-CSV "C:\My Documents\RestoreMessage.csv"; $Mailboxes | Foreach
 {Restore-RecoverableItems -Identity $_.SMTPAddress -SubjectContains Project X"
 -SourceFolder DeletedItems -FilterItemType IPM.Note}
```

It appears that Restore-RecoverableItems has similar filterable parameters to the Get-RecoverableItems cmdlet.

Practical Example 1

For the first example we need to recover any items for a user's mailbox that is not in the Deleted Items Folder. This would imply that we need to filter out any items from the Deleted Items Folder and then restore the remaining items to the mailbox:

```
Get-RecoverableItems -Identity damian@practicalpowershell.com | Where {$_.SourceFolder -ne
 'Deleted Items'} | Ft SourceFolder
```

Notice the filter based on the folder, Source Folder, and using 'Deleted Items' as the folder we do not want to see.

Practical Example 2

For this example an IT Admin accidentally removed an important email from every mailbox in an organization. The email was titled "Year In Review - 2019". It was an email, so we can assume the type of message was an IPM. Since every mailbox in the organization was affected, we can query a list of mailboxes and store their names in a variable. We can then use the max parallel.

Code

$Mailboxes = Get-ExOMailbox -Resultsize Unlimited

```
$Count = 0
$Group = 0
$ToProcess = @()
Foreach ($Mailbox in $Mailboxes) {
$ToProcess += $Mailbox.PrimarySMTPAddress
$Count++
```

```
      If ($Count -eq 10){
      $Group++
      Write-Host "Processing mailboxes to Process - Group $Group - Total - $Count"
      Restore-RecoverableItems -Identity $ToProcess -SubjectContains "Year In Review - 2019"
      -FilterItemType IPM.Note -MaxParallelSize 10
      $Count = 0
      $ToProcess = $Null
      $ToProcess = @()
      }
      }
      If (($Count -gt 0) -and ($Count -lt 10)) {
      $Group++
      Write-Host "Processing mailboxes to Process - Group $Group - Total - $Count"
      Restore-RecoverableItems -Identity $ToProcess -SubjectContains "Year In Review - 2019"
       -FilterItemType IPM.Note -MaxParallelSize 10
      }
```

When the script is run, the Restore-RecoverableItems is run against a group of 10 mailboxes at once. Since most environment's mailbox count will not be divisible by 10, any remaining mailboxes will then be processed by the section labeled 'Cleanup'. Any remainder mailbox (between 1 and 9), will be processed like the groups of 10 were processed.

Note: For either of the cmdlets, if a large number of mailboxes need to be processed, there is a parameter for that called 'MaxParallelSize'. This parameter allows for parallel processing of a large number of mailboxes that are either being queried or recovered. Valid values for this parameter are 1 to 10. This means that we can process up to 10 mailboxes at a time with these cmdlets. The parameter is not a required one.

Mailbox Diagnostics

Export-MailboxDiagnosticLogs

One of the most underused / underrated cmdlets is the Export-MailboxDiagnosticLogs. This cmdlet is an excellent source of troubleshooting information that can be used by an administrator to help find underlying issues either with a mailbox, something applying to that mailbox or some other background process that is present in Exchange Online. Let's dive right into the cmdlet's help to see what we can do with the cmdlet and then examine some practical examples that can be used as guidance.

Export-MailboxDiagnosticLogs (Get-Help -Examples)

One example is present in the Get-Help:

```
-------------------------- Example 1 --------------------------

Export-MailboxDiagnosticLogs -ComponentName CalendarPermissions -Identity "Yuuto Sasaki"

This example retrieves the calendar permissions diagnostic log for the mailbox named Yuuto
Sasaki.
```

What we see are two building blocks for our one-liners - Component and Identity. We also have ExtendedProperties and Archive as switches we can utilize as well. Let's start with components that we can chose for the cmdlet:

BirthdayAssistant	CalendarPermissions	CalendarSharingLocalFolder
DefaultViewIndexer	FreeBusyPublishingAssistantQuickLog	HoldTracking
InternetCalendar	MRM	OOFRules
RemindersAssistant	SharingMigrationAssistant	SharingSyncAssistant

Now what is interesting is one statement in the Get-Help for the cmdlet:

```
-ComponentName <String>
    The ComponentName parameter specifies the component that you want to retrieve the
    diagnostic logs for. Valid values depend on the type and location of the mailbox
    (on-premises or Exchange Online). Possible values are:
```

This could be interpreted simply that not all options are available for both Online and on-premises so expect differences. However, what is interesting, is that Exchange Online has even more options than those that are listed.

We can see what is available by choosing an option that is not available as the error message display valid options. Available options also can vary among mailboxes on the same platform. Sample error from Exchange Online when we chose the wrong options:

```
Logs for component 'OOFRules' weren't found in mailbox 'Nicholas Scoles'. Available logs: 'MRM,
DefaultViewIndexer, SweepRules, OnlineMeetings, SharingMigrationAssistant,
FreeBusyPublishingAssistantQuickLog, InternalCalendarSharingMigration, BirthdayAssistant,
SharingSyncAssistant, RemindersAssistant, CalendarPermissions, MeetingMessageProcessingAgent,
OOF,HoldTracking,SubstrateHoldTracking'
    + CategoryInfo          : NotSpecified: (:) [Export-MailboxDiagnosticLogs], ObjectNotFoundExce
   ption
```

Exchange Server 2016 Error Message:

```
Logs for component 'SubstrateHoldTracking' weren't found in mailbox 'Administrator'. Available
logs: 'BirthdayAssistant, InternalCalendarSharingMigration, SharingSyncAssistant,
CalendarPermissions, RemindersAssistant, OOF,HoldTracking'
    + CategoryInfo          : NotSpecified: (:) [Export-MailboxDiagnosticLogs], ObjectNotFoundE
   ption
```

In the end, if we review both Exchange 2016's error and Exchange Online, we see we have the following options (differences are highlighted):

ExO	Exchange 2016
BirthdayAssistant	BirthdayAssistant
CalendarPermissions	CalendarPermissions
DefaultViewIndexer	HoldTracking
FreeBusyPublishingAssistantQuickLog	InternalCalendarSharingMigration
HoldTracking	OOF
InternalCalendarSharingMigration	RemindersAssistant
MeetingMessageProcessingAgent	SharingSyncAssistant
MRM	
OnlineMeetings	
OOF	
RemindersAssistant	
SharingMigrationAssistant	
SharingSyncAssistant	
SubstrateHoldTracking	
SweepRules	

As we can see, Exchange Online has more Diagnostic logs to pull from for each mailbox that exists. Now let's explore these Components to see what exactly we can retrieve from them.

HoldTracking

In this example we can review two mailboxes, one with a hold and one without. The one without the hold will look like this:

```
RunspaceId   : 317840fa-429f-4611-b860-6a39f8961914
MailboxLog   : []
LogName      : HoldTracking
Identity     : [redacted]
IsValid      : True
ObjectState  : Unchanged
```

The below mailbox, however, has had holds applied to it over time. We can see the various dates in the mailbox log:

```
RunspaceId   : 317840fa-429f-4611-b860-6a39f8961914
MailboxLog   : [{"ed":"2018-07-17T17:02:34.0629748Z","hid":"mbx7637cba550264390a75a05c023ee54cf:1","
               ht":1,"lsd":"2018-07-12T06:48:53.4414718Z","osd":"2018-07-12T06:48:53.4414718Z"},{"ed
               ":"2019-10-17T07:24:22.1836964Z","hid":"mbx3c9b56729b6d49d0805faee53e48d11c:3","ht":1
               ,"lsd":"2018-07-17T17:02:34.0629748Z","osd":"2018-07-17T17:02:34.0629748Z"},{"ed":"20
               19-11-16T12:20:27.9557492Z","hid":"DelayReleaseHold","ht":3,"lsd":"2019-10-17T07:24:2
               2.1836964Z","osd":"2018-07-17T17:02:34.0629748Z"},{"ed":"2019-10-17T07:24:22.1836964Z
               ","hid":"mbxdd3ef3b2e11943ee9f6fc220b80cfd6c:2","ht":1,"lsd":"2018-07-18T19:00:14.679
               1062Z","osd":"2018-07-18T19:00:14.6791062Z"},{"ed":"2019-10-17T07:24:22.1836964Z","hi
               d":"UniH0c70eac9-adfb-4574-b66f-c41ff4df80a8","ht":1,"lsd":"2018-07-19T20:17:20.17475
               01Z","osd":"2018-07-19T20:17:20.1747501Z"},{"ed":"2019-09-12T04:28:44.3273150Z","hid"
               :"mbx32e1c6a402e0443abc824ef3d98da64f:1","ht":4,"lsd":"2018-11-22T18:40:17.8742589Z",
               "osd":"2018-11-22T18:40:17.8742589Z"},{"ed":"2019-09-12T04:28:44.3273150Z","hid":"mbx
               [redacted]
```

ExtendedProperties

This one switch can provide a veritable goldmine of details on a mailbox. Keep in mind that the data is stored in the XML format so the default output may not be easy to work with. This, however, can be changed to make the values more accessible. An Example:

```
([xml](Export-MailboxDiagnosticLogs -Identity damian@practicalpowershell.com
 -ExtendedProperties ).MailboxLog).Properties.MailboxTable.Property
```

Why is the one-liner above formatted this way? If we look at the output of this one-liner without some of the drilling down, we have a core one-liner that looks like this:

```
(Export-MailboxDiagnosticLogs -Identity damian@practicalpowershell.com -ExtendedProperties)
```

Which provides this output:

```
RunspaceId   : 317840fa-429f-4611-b860-6a39f8961914
MailboxLog   : <Properties>
                 <MailboxTable>
                   <Property>
                     <Name>DatabaseSchemaVersion</Name>
                     <Value>65563</Value>
                   </Property>
                   <Property>
```

If we combine those results with the previous cmdlet, we can see these connections:

```
([xml](Export-MailboxDiagnosticLogs -Identity damian@practicalpowershell.com -ExtendedProperties).MailboxLog).Properties.MailboxTable.Property

                    RunspaceId   : 317840fa-429f-4611-b860-6a39f8961914
                    MailboxLog   : <Properties>
                                     <MailboxTable>
                                       <Property>
                                         <Name>DatabaseSchemaVersion</Name>
                                         <Value>65563</Value>
                                       </Property>
```

When we run the cmdlet, with the [XML] translation code, we see these results (which look like a Format Table for a PowerShell object).

Some interesting items that can be located in this view:

```
Name                                                Value
----                                                -----
DatabaseSchemaVersion                               65563
StorageQuotaLimit                                   102760448
NormalMessageSize64                                 93790296
SystemMessageSize                                   12712193
SystemMessageSizeShutoffQuotaAuto                   26843545600
TotalPages                                          2580
PersistableTenantPartitionHint                      0x4EC50C5D8200B84EA300CE17A036F3F4
IsContentIndexingEnabled                            True
ProhibitReceiveQuota                                104857600
ProhibitSendQuota                                   103809024
Size                                                95023739
SystemMessageCount                                  2877
ExtendedSize                                        95023739
ExtendedDumpsterSize                                81197
MaxSubmitMessageSize                                35840
MaxMessageSize                                      36864
MailboxMiscFlags                                    0
MailboxGuid                                         0x418066161CCCF0624EBC68E1D07B8C50BA
```

ELCLastSuccessTimestamp - import as this property is changed when the mailbox has been processed by the managed folder assistant.
OofScheduleStart - start of the user's Out of Office notification.
OofScheduleEnd - end of the user's Out of Office notification.
Size - current mailbox size.
StorageQuotaLimit - current max size of a mailbox.

What else can we use the Mailbox Diagnostics log for?

Sweep Rules:

```
Export-MailboxDiagnosticLogs -Identity damian@domain.com -ComponentName sweeprules
```

```
RunspaceId  : 317840fa-429f-4611-b860-6a39f8961914
MailboxLog  : 1/13/2020 11:33:26 AM, Mailbox: 'Damian Scoles', Entry No sweep rules to process.,

              1/13/2020 11:33:25 AM, Mailbox: 'Damian Scoles', Entry Executing cleanup actions for DisplayName='Damian Scoles';
              SmtpAddress=Damian@scolesfamily.net; ActivityId=ca9f611c-3c44-4c37-8c59-8c7947586635,

              1/12/2020 9:17:57 AM, Mailbox: 'Damian Scoles', Entry No sweep rules to process.,

              1/12/2020 9:17:57 AM, Mailbox: 'Damian Scoles', Entry Executing cleanup actions for DisplayName='Damian Scoles';
              SmtpAddress=Damian@scolesfamily.net; ActivityId=2ae95cfc-cc93-449e-b335-d373431225e4,

              1/11/2020 5:46:01 AM, Mailbox: 'Damian Scoles', Entry No sweep rules to process.,

              1/11/2020 5:46:00 AM, Mailbox: 'Damian Scoles', Entry Executing cleanup actions for DisplayName='Damian Scoles';
              SmtpAddress=Damian@scolesfamily.net; ActivityId=d2e468a2-35f4-414a-9af9-ed3553b4dfed,
```

Calendar Entries

```
Export-MailboxDiagnosticLogs -Identity damian@practicalpowershell.com -ComponentName Calendar
```

```
RunspaceId  : 317840fa-429f-4611-b860-6a39f8961914
MailboxLog  : 1/14/2020 1:42:26 PM, Mailbox: 'Damian Scoles', Entry Meeting request type is full
              update, reason: Item is not responded. Last rolling highlight: 139. New highlight:
              139. InternetMessageId:
              <CWLP123MB23568234A2FD76C65EAF437FDD340@CWLP123MB2356.GBRP123.PROD.OUTLOOK.COM>.
              Goid: 040000008200E00074C5B7101A82E008000000008ED123EB0BCAD501000000000000000001000000
              00C7DB534E724A140AF98CB670A3D92D9

              10/22/2019 6:33:17 AM, Mailbox: 'Damian Scoles', Entry Meeting request type is full
              update, reason: Item is not responded. Last rolling highlight: 147. New highlight:
              147. InternetMessageId:
              <BN8PR00MB05958746BD3DE35C6EE5CC30B7680@BN8PR00MB0595.namprd00.prod.outlook.com>.
              Goid: 040000008200E00074C5B7101A82E00800000000010534EC69C88D501000000000000000001000000
              0A2E9573A73886A4594D457F37B82E6F2
```

Calendar Permissions

```
Export-MailboxDiagnosticLogs -Identity damian@practicalpowershell.com -ComponentName
 CalenderPermissions
```

```
RunspaceId  : 7270b435-8fb9-4c78-a2ed-edb1c71021ab
MailboxLog  : 1/17/2020 11:21:15 AM, Mailbox: [redacted], Entry Client Type: Migration Move
              Status:MailboxMoveSucceeded Permission Table:
              Folder Name: Calendar Security Descriptor: S-1-5-18
              Anonymous Permission:None Member Rights:None
              Default Permission:Custom Member Rights:FreeBusySimple

              10/17/2019 4:05:34 AM, Mailbox: [redacted], Entry Client Type: Migration Move
              Status:MailboxMoveSucceeded Permission Table:
              Folder Name: Calendar Security Descriptor: S-1-5-18
              Anonymous Permission:None Member Rights:None
              Default Permission:Custom Member Rights:FreeBusySimple

              8/19/2018 5:33:38 PM, Mailbox: [redacted], Entry Client Type: Migration Move
              Status:MailboxMoveSucceeded Permission Table:
```

TimeProfile

```
Export-MailboxDiagnosticLogs -Identity damian@practicalpowershell.com -ComponentName
 TimeProfile
```

```
RunspaceId  : 317840fa-429f-4611-b860-6a39f8961914
MailboxLog  : 1/14/2020 3:43:21 AM, Mailbox: 'Damian-PP', Entry
              2020-01-14T03:43:21.4049347Z: Request id "c928dc66-2e51-48b9-b0b6-f2a037f84690"
              2020-01-14T03:43:21.4049347Z: -------------- SetUserSettings started --------------
              2020-01-14T03:43:21.4179261Z: Time format not found. Using defaults
              2020-01-14T03:43:21.4359149Z: Server name CY4PR2201MB1704, build 16.01.2504.000,
              Mailbox capabilities: None, Current mode: Griffin
              2020-01-14T03:43:21.4359149Z: Saving item FoodEstablishmentReservation
              ProcessingLevel Calendar FRE False
              2020-01-14T03:43:21.4628987Z: Settings are saved to SDS
              2020-01-14T03:43:21.4878833Z: Autoblocker state: Budget:298812/300000
              NextAllocation: 2020-01-14 13:31:55Z Policy 302178
              2020-01-14T03:43:21.4878833Z: -------------- SetUserSettings ended --------------
```

Inactive Mailboxes

Inactive Mailboxes are mailboxes that usually were attached to a former employee where the mailbox was put on hold and then the Exchange Online license (or all Office 365 licenses) were removed from the user. The mailbox then is removed from Exchange as a regular mailbox and becomes a discoverable mailbox for Office 365. How can we find these mailboxes? Get-ExOMailbox.

The Get-ExOMailbox cmdlet contains two different parameters that can be used to reveal these mailboxes:

```
-InactiveMailboxOnly [<SwitchParameter>]
    The InactiveMailboxOnly switch specifies whether to return only inactive mailboxes in the results. You don't
    need to specify a value with this switch.

    An inactive mailbox is a mailbox that's placed on Litigation Hold or In-Place Hold before it's soft-deleted.
    The contents of an inactive mailbox are preserved until the hold is removed.

    To return both active mailboxes and inactive mailboxes in the results, don't use this switch. Instead, use the
    IncludeInactiveMailbox switch.
-IncludeInactiveMailbox [<SwitchParameter>]
    The IncludeInactiveMailbox switch specifies whether to include inactive mailboxes in the results. You don't
    need to specify a value with this switch.

    An inactive mailbox is a mailbox that's placed on Litigation Hold or In-Place Hold before it's soft-deleted.
    The contents of an inactive mailbox are preserved until the hold is removed.

    To return only inactive mailboxes in the results, don't use this switch. Instead, use the InactiveMailboxOnly
    switch.
```

The difference between the two is that one will include Inactive Mailboxes when you are performing actions against mailboxes. The other switch will show ONLY Inactive Mailboxes and exclude all others. Which switch that is chosen is determined by what your end goal is with both Inactive Mailboxes and possibly user mailboxes. Let's review a couple of examples to see the differences in action.

Example 1

For this scenario, we have an HR department that would like a report on all mailboxes, as well as any that were designated as Inactive for discovery purposes. The request is for the name of the user with discovery properties:

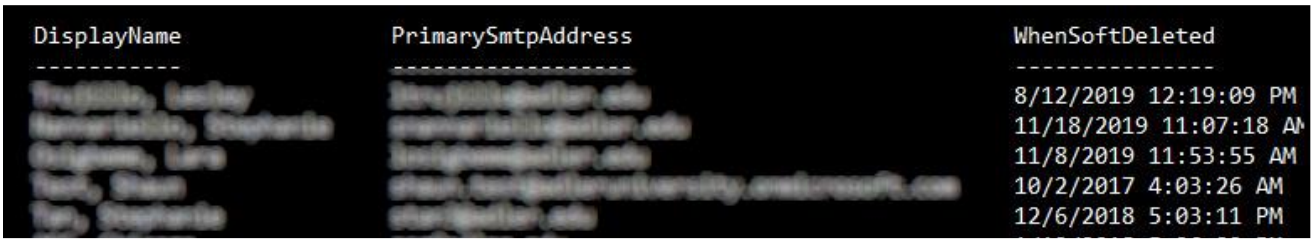
```
Get-ExOMailbox -PropertySets All -IncludeInactiveMailbox -ResultSize Unlimited | ft
 DisplayName, RetentionPolicy, LitigationHoldDuration, IsInactiveMailbox -Auto
```

Example 2

Your IT manager would like a report of ONLY Inactive Mailboxes for reporting purposes and would like to see how many are there and when the mailbox was originally removed from Exchange Online. Luckily one of the properties on mailboxes is 'WhenSoftDeleted'.

Method 1

```
Get-ExOMailbox -InactiveMailboxOnly -PropertySets Minimum,SoftDelete | Ft DisplayName,
 PrimarySMTPAddress, WhenSoftDeleted
```



Method 2
While this method does NOT use the -InactiveMailboxesOnly switch, it basically performs the task. Basically we are querying all mailboxes, including Inactive Mailboxes and then filtering for the IsInactiveMailbox property and pulling out only those that have that value set to $True.

Get-ExOMailbox -PropertySets Minimum, SoftDelete -IncludeInactiveMailbox -ResultSize Unlimited -Filter { IsInactiveMailbox -eq $True} | ft DisplayName, IsInactiveMailbox

As we can see, Inactive Mailbox information can be pulled.

Note: When querying for Inactive Mailboxes, we also can pull Soft Deleted mailboxes. There are switches for both Get-ExOMailbox and Get-ExOMailboxStatistics for Soft Deleted mailboxes. It cannot however clear out any errors like the mailbox is not unique. Further filter and/or error handling may be needed to handle this scenario:

```
The specified mailbox "testemail2[redacted]  isn't unique.
    + CategoryInfo          : NotSpecified: (:) [Get-MailboxStatistics], ManagementObjectAmbiguous
   Exception
    + FullyQualifiedErrorId : [Server=BN8PR03MB4802,RequestId=3cbf95d1-e2f2-4bee-b032-8dab391afad5
   ,TimeStamp=1/14/2020 5:59:06 AM] [FailureCategory=Cmdlet-ManagementObjectAmbiguousException] E
  1DBA96A,Microsoft.Exchange.Management.MapiTasks.GetMailboxStatistics
    + PSComputerName        : ps.outlook.com
```

Mailbox Holds

Mailbox holds are useful if your organization needs a way to preserve emails for discovery or for recovery purposes. There are a number of Hold Types that are included in Office 365. These holds can help your legal teams as well as your IT teams. In this section we will briefly review the types of holds and how to determine which hold has been applied to a mailbox.

Hold Types?

If we review the properties of a mailbox, we can see that we have the following Hold Types for mailboxes:

```
LitigationHoldEnabled     : False
RetentionHoldEnabled      : False
EndDateForRetentionHold   :
StartDateForRetentionHold :
LitigationHoldDate        :
LitigationHoldOwner       :
ComplianceTagHoldApplied  : False
DelayHoldApplied          : False
DelayReleaseHoldApplied   : False
LitigationHoldDuration    : Unlimited
SCLDeleteThreshold        :
SCLRejectThreshold        :
SCLQuarantineThreshold    :
SCLJunkThreshold          :
InPlaceHolds              : {}
```

What we see is that we have these holds:

Litigation Hold: Does the mailbox have a litigation hold in place.
Retention Hold: Was a Retention Hold created and applied to this mailbox.
Compliance Tag Hold: Means there is a hold, probably from the Security and Compliance Center.
Delay Hold: Hold of some type of another. In case, if the ComplianceTagHoldApplied is True, so is this value.
Delay Release Hold: Non-mailbox data that has a hold placed on it, like Yammer, Teams, etc.

Lastly, we see that there is a value called 'InPlaceHolds'. This value is used to display the hold that is applied to a mailbox. If we want, we can find all mailboxes that have a hold applied, by looking for a non-null value for the InPlaceHolds property:

Get-ExOMailbox -PropertySets Hold,Retention -ResultSize Unlimited | Where {$_.InPlaceHolds -ne $Null} | Ft *hold*

Now, how do we interpret any results we get? Let's use some sample mailboxes to determine the holds and what they mean. In the below screenshots we have two different mailboxes, both show Litigation Holds, but the InPlaceHolds value is different:

```
InPlaceHolds                            LitigationHoldEnabled RetentionHoldEnabled ComplianceTagHoldApplied DelayHold
------------                            --------------------- -------------------- ------------------------ ---------
{UniHd4fa93d4-993f-47ea-9eb0-d9d322112fb6}                True                False                    False
```

Vs.

```
InPlaceHolds                        LitigationHoldEnabled RetentionHoldEnabled ComplianceTagHoldApplied DelayHoldApplied
------------                        --------------------- -------------------- ------------------------ ---------------
{368a5445ba514d9499fb6fdc62cb5af0}                  True                False                    False           False
```

Notice the name of the InPlaceHolds value, one starts with UniH and the other starts with numbers. How can we interpret these values?

See this Link:
https://docs.microsoft.com/en-us/office365/securitycompliance/identify-a-hold-on-an-exchange-onlinemailbox

For your mailboxes in Office 365, we typically see two types of holds 'MBX' and 'UniH' holds. Checking the link about, here is what we find out about these holds:

> **UniH** - This is what is known as an 'eDiscovery hold'. This kind of hold can be triggered in the Security and Compliance Center as well (I believe, but it was not in your case) from a eDiscovery Hold in Exchange Online.
> **MBX** - 'Office 365 retention policy specifically applied to the mailbox' - In your case, it was your retention tags/labels that are configured in the Security and Compliance Center.

Now that we've found the holds, how do we identify their purpose and either document them or remove them from the mailbox.

- Connect to the Security and Compliance Center PowerShell: https://docs.microsoft.com/en-us/powershell/exchange/office-365-scc/connect-to-scc-powershell/mfa-connect-to-scc-powershell
- Once connected, list all the Compliance Hold Policies - Get-HoldCompliancePolicy

Remove Case Hold Compliance Policy

If we want to remove a particular mailbox from a policy, we can do this:

Set-RetentionCompliancePolicy <Policy Name> -AddExchangeLocationException <email address>

Once these changes are made, it can take some time for these to apply. Sometimes as little as an hour or two, sometimes up to a week. These are supposed to be changed by the Managed Folder Assistant (MFA) but have some confidence in that in the case of multiple rollbacks, the MFA had run and the tags were not removed. So, patience is key here.

UniH InPlaceHolds

Now, for the UniH InPlaceHold, these are harder to work with, but only because of the correct permissions as by default you may not have access to the case in the eDiscovery part of the Security and Compliance Center. In order to get to these

cases, we need to grant ourselves the eDiscovery Administrator role (not Manager – not sufficient rights). After this is done, we should be able to list the eDiscovery Cases involved. The key thing to note, in SCC PowerShell, this hold is known as a Case Hold Policy and can be found using the GUIDS found in the 'InPlaceHolds' value on the mailbox. We need the GUID of the hold, which is the value we found, minus the 'UniH' prefix, which means 'd4fa93d4-993f-47ea-9eb0-d9d322112fb6':

Get-CaseHoldPolicy d4fa93d4-993f-47ea-9eb0-d9d322112fb6

This gives us the 'Migrated Users' Case Hold Policy. Then we can exclude the user (RemoveExchangeLocation). How to remove the user from the eDiscovery case:

Set-CaseHoldPolicy d4fa93d4-993f-47ea-9eb0-d9d322112fb6 -RemoveExchangeLocation <email address>

Now you need to wait for replication like the previous section.

Note: A Mailbox Search in Exchange Online can also cause this to show. For your tenant, I would not create any as the SCC has these capabilities and more. If you cannot find a Case Hold Policy, then look for a Mailbox Search in Exchange Online PowerShell. Once a hold has been removed, you will see a '-' appear in front of the InPlaceHolds type like so:

This is the exclusion/exception being displayed on the mailbox.

Now, one thing that could happen is that if there are two settings that could 'stick' if we were not patient enough. Those were the aforementioned 'ComplianceTagHoldApplied' set to True and 'DelayHoldApplied' set to True as well. Some mailboxes would eventually see the 'ComplianceTagHoldApplied' change to False, but the 'DelayHoldApplied' would still be true. Here is how to resolve the second issue:

```
Set-Mailbox <email address> -RemoveDelayHoldApplied
```

For reference:
https://docs.microsoft.com/en-us/powershell/module/exchange/mailboxes/set-mailbox?view=exchange-ps

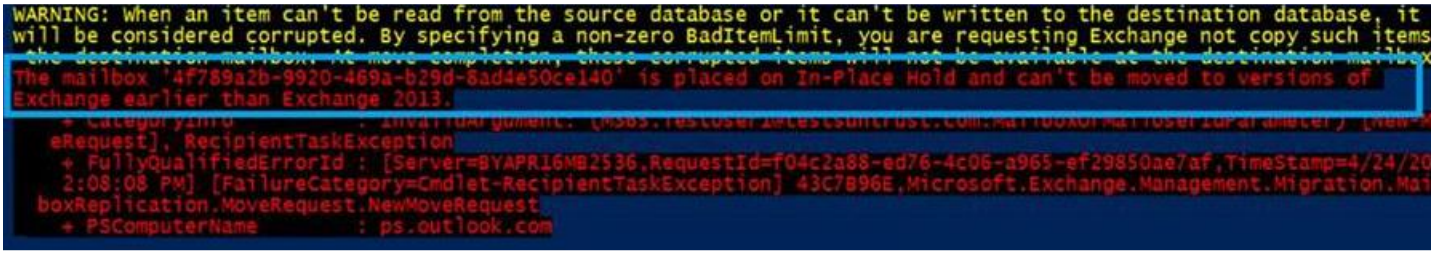
-RemoveDelayHoldApplied
This parameter is available only in the cloud-based service.

The RemoveDelayHoldApplied switch specifies whether to remove delay holds from the mailbox. You don't need to specify a value with this switch.

The removal of a hold from a mailbox is temporarily delayed to prevent the accidental purge of content that's no longer affected by the hold. This temporary delay in the removal of the hold is known as a delay hold. To see the hold history on a mailbox, replace <MailboxIdentity> with the name, email address, or alias of the mailbox, and run this command: Export-MailboxDiagnosticLogs -Identity <MailboxIdentity> -ComponentName HoldTracking. Now, the last issue we may run into is that if a mailbox has a 'ComplianceTagHoldApplied' it could stay true no matter what the other settings were and even if exclusions were in place. If that happens, a support call may be needed to ask Microsoft to remove it for us on the back end. They will make you ask them in writing (email) to do so. It takes time and they do not guarantee a time as to when it gets completed.

Exchange 2010 Note

While most organizations have moved on from Exchange 2010, there are some companies ranging from small to very large that still have this installed. What we find is that if we are moving mailboxes to Exchange Online and mailboxes needs to be moved back due to issues on the end user side, there are some possible issues with Holds. Specifically, if a mailbox in Exchange Online has a hold applied to and the mailbox needs to be moved back to Exchange 2010, then the holds need to be removed. If you do not, then an error message can be generated in PowerShell that looks like this:



Searching Mailboxes

When the First Edition of this book was written, we had many options for searching mailboxes - *MailboxSearch cmdlets, Search-Mailbox, EWS and the Security and Compliance Center. However, Microsoft recently reinforced their retirement message for the first two PowerShell cmdlet sets:

```
WARNING: Warning: As of April 1, 2020, the *-MailboxSearch cmdlets are being retired. You will no
longer be able to create new searches using the New-MailboxSearch cmdlet and Microsoft Support will
 no longer provide assistance as per the notice in the link below.  You may continue to use these
cmdlets at your own risk.  On July 1, 2020, Start-MailboxSearch, Stop-MailboxSearch, and
Set-MailboxSearch won't be available. Only Remove-MailboxSearch will be available to delete
searches and holds. See https://go.microsoft.com/fwlink/?linkid=2113221 to learn more.
```

```
WARNING: WARNING:  On April 1, 2020, the Search-Mailbox cmdlet is being retired and Microsoft
Support will no longer provide assistance.  See https://go.microsoft.com/fwlink/?linkid=2113221 to
learn more.
```

We see that both of these have a Microsoft link. This link leads to this page:

https://docs.microsoft.com/en-us/microsoft-365/compliance/legacy-ediscovery-retirement

Microsoft uses this landing page to basically describe the retirement of all of these PowerShell cmdlets and what this means to admins. In the end, it means that Exchange Online is losing more PowerShell cmdlets and these operations are moving over to the Security and Compliance Center. In order to perform searches, Microsoft has a series of cmdlets that we should use now to replace those operations:

```
*-ComplianceSearch
*-ComplianceSearchAction
*-ComplianceCase
```

Following this page we will dive into these cmdlets some more in the Security and Compliance Center. Compliance Search In Chapter 9 we covered multiple topics and examples for compliance related items for Exchange Online - DLP, Journaling and Rights Management. On the prevoius page we reviewed searching in Exchange Online. However, Microsoft has now included some cmdlets that were only either in Exchange Online or the Security and Compliance Center. These cmdlets are specific to Compliance Searches in the Security and Compliance Center. These cmdlets are the recommended replacement for *-MailboxSearch cmdlets that are being deprecated. Let's start with what cmdlets are available and then see what we can do with them in Exchange.

PowerShell

```
Get-Command *ComplianceSearch*
```

Which provides us with these cmdlets:

```
Get-ComplianceSearch                    Get-ComplianceSearchAction
Get-ComplianceSearchActionStatistics    Invoke-ComplianceSearchActionStep
New-ComplianceSearch                    New-ComplianceSearchAction
Remove-ComplianceSearch          Set-ComplianceSearch
Set-ComplianceSearchAction       Start-ComplianceSearch
Stop-ComplianceSearch
```

By default we should have no Compliance Searches or Compliance Search Actions.

Requirements

In order to perform a Compliance Search we need to meet the following requirements:

Security - Assigned the 'Mailbox Search' management role.

```
New-ManagementRoleAssignment -Name "Exchange Servers_John" -Role "Exchange Servers" -User John
```

Start the Search - Once a Compliance Search is created, it has to be started as well.

Apply an Action - Determine what is to be done with the search results.

New-ComplianceSearch

Well, let's go ahead and start creating Compliance. Since we are new to the cmdlet, we can review Get-Help to see if we have any relevant examples to work with:

```
Get-Help New-ComplianceSearch -Examples
```

```
------------------------- Example 1 -------------------------

New-ComplianceSearch -Name "Hold Project X" -ExchangeLocation "Finance Department"

------------------------- Example 2 -------------------------

New-ComplianceSearch -Name "Hold-Tailspin Toys" -ExchangeLocation "Research Department" -ContentMatchQuery
"'Patent' AND 'Project Tailspin Toys'"
```

If we look at the available parameters for this cmdlet we can see we have some options to choose from:

```
AllowNotFoundExchangeLocationsEnabled  ContentMatchQuery
ExchangeLocation                       ExchangeLocationExclusion
Force                                  Language
```

Note: Content queries can be constructed with the Keyword Query Language (KQL) which can be referenced here: https://docs.microsoft.com/en-us/sharepoint/dev/general-development/keyword-querylanguage-kql-syntax-reference

Example One - Specific Word Query

Your Compliance Administrator just received notice that a few people in Research and Development (RnD) are involved in a patent dispute and specifically she needs to search emails of a group of users. She hands you a list of employees in RnD that are to be searched. Keywords for the search are 'Patent', 'US 8,965,465 B2' and 'SmartPhone'. All of the searches need to be associated with a case called 'SmartPhone Patent Dispute'.

First, we need to read the CSV file in and store it in a variable called $CSV. The file is stored on a server called FS01 and the Compliance Administrator provides the entire path to be used:

```
$CSV = Import-CSV '\\fs01\Compliance\SmartphonePatentDispute\RnDUserList.csv'
```

The case name is also stored in a new variable called $Case:

```
$Case = 'SmartPhone Patent Dispute'
```

Then, using the criteria provided from the Compliance Administrator, we can store that in a variable to be used later as well:

```
$Criteria = "'Patent' AND 'US 8,965,465 B2' AND 'SmartPhone'"
```

We can then use a Foreach loop to process each name in the CSV, create a search for each user:

```
Foreach ($Line in $CSV) {
$User = $Line.User
$Name = "$Case - $User"
New-ComplianceSearch -Name $Name -ExchangeLocation $User -ContentMatchQuery $Criteria
}
```

We can then verify these cases are created with 'Get-ComplianceSearch':

```
Name                                                                RunBy  JobEndTime Status
----                                                                -----  ---------- ------
SmartPhone Patent Dispute - PeteBlanket@practicalpowershell.com                       NotStarted
SmartPhone Patent Dispute - Sam@practicalpowershell.com                               NotStarted
SmartPhone Patent Dispute - Lance@practicalpowershell.com                             NotStarted
SmartPhone Patent Dispute - Ann.Ples@practicalpowershell.com                          NotStarted
```

Once completed we can check our results with a couple of cmdlets:

```
Get-ComplianceSearch | Fl
```

```
Language              :
StatusMailRecipients  : {}
LogLevel              : Suppressed
IncludeUnindexedItems : True
ContentMatchQuery     : 'Patent' AND 'US 8,965,465 B2' AND 'SmartPhone'
SearchType            : EstimateSearch
HoldNames             : {}
SearchNames           : {}
RefinerNames          : {}
Region                :
Refiners              :
Items                 : 0
Size                  : 0
UnindexedItems        : 0
UnindexedSize         : 0
SuccessResults        : {}
SearchStatistics      :
Errors                :
ErrorTags             : {}
NumFailedSources      : 0
JobId                 : c9fa1f24-6193-4e1e-84b2-08d7428922b0
Name                  : SmartPhone Patent Dispute - PeteBlanket@practicalpowershell.com
```

Notice on the previous page that all of the searches show a Status of 'NotStarted'. We can use Start-ComplianceSearch to start them.

```
Get-ComplianceSearch | where {$_.Status -eq 'NotStarted'} | Start-ComplianceSearch
```

After some time, the searches will complete:

```
Name                                                                RunBy         JobEndTime              Status
----                                                                -----         ----------              ------
SmartPhone Patent Dispute - PeteBlanket@practicalpowershell.com Administrator 9/26/2019 2:03:56 PM  Completed
SmartPhone Patent Dispute - Sam@practicalpowershell.com         Administrator 9/26/2019 2:03:53 PM  Completed
SmartPhone Patent Dispute - Lance@practicalpowershell.com       Administrator 9/26/2019 2:03:58 PM  Completed
SmartPhone Patent Dispute - Ann.Ples@practicalpowershell.com    Administrator 9/26/2019 2:03:59 PM  Completed
```

New-ComplianceSearchAction

What about this cmdlet New-CompilanceSearchAction? Compliance Search Actions determine what action the search performs (Preview, Export, etc.):

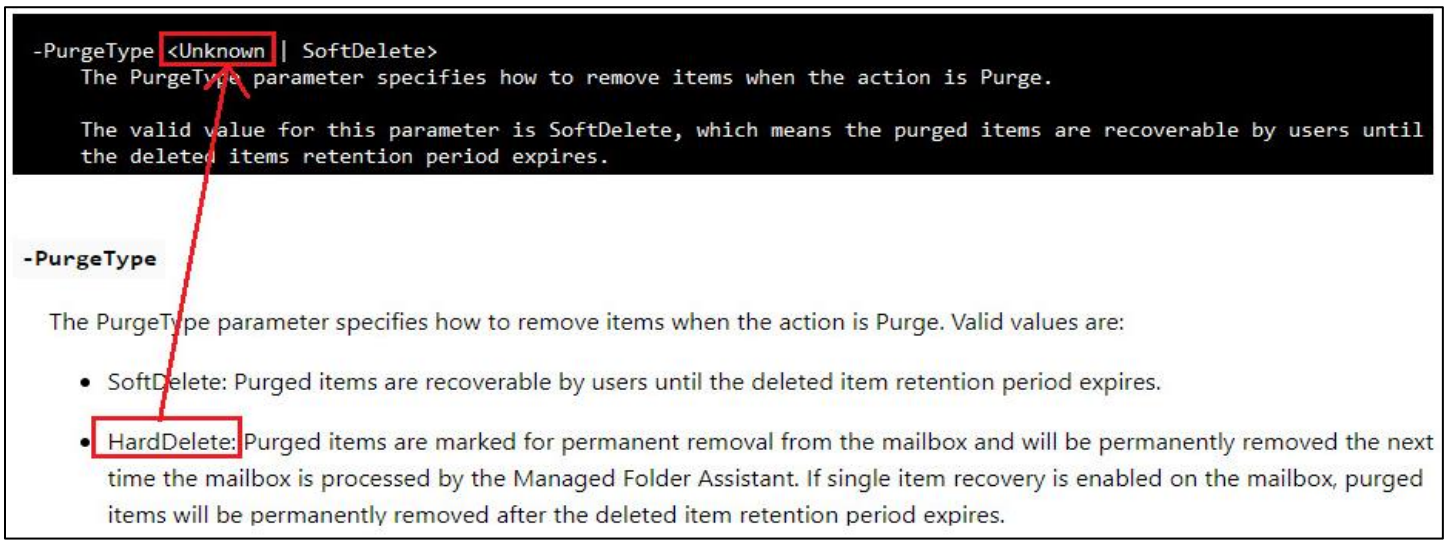
Get-Help New-ComplianceSearchAction -Examples

```
-------------------------- Example 1 --------------------------

New-ComplianceSearchAction -SearchName "Project X" -Preview

This example creates a preview search action for the compliance search named Project X.

-------------------------- Example 2 --------------------------

New-ComplianceSearchAction -SearchName "Project X" -Export

This example creates an export search action for the compliance search named Project X.

-------------------------- Example 3 --------------------------

New-ComplianceSearchAction -SearchName "Remove Phishing Message" -Purge -PurgeType SoftDelete
```

This cmdlet would be used after a search was created. One interesting note is that the -PurgeType option seems corrupt in the Get-Help for New-ComplianceAction as shown below:

```
-PurgeType <Unknown | SoftDelete>
    The PurgeType parameter specifies how to remove items when the action is Purge.

    The valid value for this parameter is SoftDelete, which means the purged items are recoverable by users until
    the deleted items retention period expires.
```

If we review documentation that Microsoft provides online, we find that the missing or 'unknown' type of Purge is actually 'HardDelete' which would make sense as it pairs well with the SoftDelete that we see:



Example - Phishing Email Removal
In this example your organization has received a series of Phishing emails that have been delivered to everyone's mailbox. Each mailbox has 10 copies of the same messages. We need to be able to clean up all of these messages, without any user interaction. What can we do?

First, we need to construct a Compliance Search. This search will not be connected to a Compliance Case as we don't need this feature.

```
$Name = 'Phishing Email Search'
$Criteria = "subject:'Wire Transfer Request'"
New-ComplianceSearch -Name $Name -ExchangeLocation All -ContentMatchQuery $Criteria
```

Once this is created, we can then start the search:

```
Start-ComplianceSearch -Identity $Name
```

Once the search is started, we can now create an action. Since these are Phishing emails, we need to remove them and not allow the end user any access to the emails once we complete this task. The New-ComplianceSearchAction has a couple of options we can utilize - Purge and PurgeType. The Get-Help description for the 'PurgeType' parameter is a bit deceptive as only one option is available and the other option listed is not correct as it is called 'Unknown'. This value should be 'HardDelete' according to online help for the cmdlet.

We need to run the Compliance Search Action with the same name as the Compliance Search. We will also use both purge actions:

```
New-ComplianceSearchAction -SearchName $Name -Purge -PurgeType SoftDelete
```

Make sure this is what you want to do before hitting yes to this question:

```
PS C:\> New-ComplianceSearchAction -SearchName $Name -Purge -PurgeType SoftDelete

Confirm
Are you sure you want to perform this action?
This operation will make message items meeting the criteria of the compliance search "Phishing Email Search" completely
inaccessible to users. There is no automatic method to undo the removal of these message items.

[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [?] Help (default is "Y"):
```

Once this starts, any matching items will be SoftDeleted from the mailbox. Now, what if we needed to modify the search?

Set-ComplianceSearch
Once we have Compliance Searches created, we can manipulate some of their details. In order to do so, we need to utilize the 'Set-ComplianceSearch' cmdlet. Let's review what we can do with this cmdlet:

```
Get-Help Set-ComplianceSearch -Examples
```

```
-------------------------- Example 1 --------------------------

Set-ComplianceSearch -Identity "Project X" -ExchangeLocation All

This example changes the existing compliance search named Project X. The scope of the Exchange search is changed
to all mailboxes.
```

```
------------------------- Example 2 -------------------------

Set-ComplianceSearch -Identity "Contoso Case Search 1" -HoldNames All -ExchangeLocation $null -SharePointLocation
$null

This example changes an existing compliance search that's associated with an eDiscovery case in the Office 365
Security & Compliance Center. The scope of the search is changed from searching selected mailboxes and SharePoint
sites to searching all content locations that have been placed on hold in the eDiscovery case.
```

One possible use would be changing the language of the search when it was realized that the mailboxes within scope were not native English speakers.

```
Set-ComplianceSearch -Identity $Name -Language fr-ch
```

The above would then change the search language to French (Switzerland) from US (English).

Email AutoForward Protection

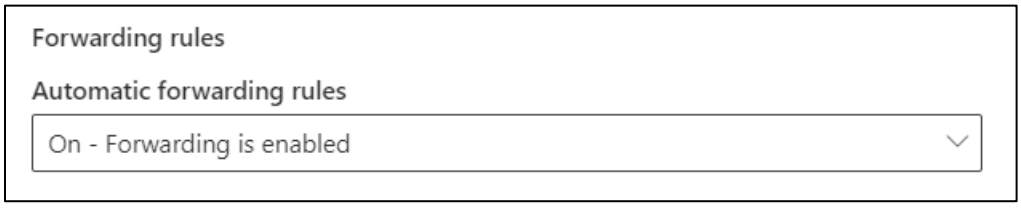
Recently Microsoft introduced some changes to how AutoForwarded messages are handled in Exchange Online. As such it seems appropriate to work through those changes in this book. AutoForwarding by itself is not necessarily a sign of malicious activity or a bad actor getting access to your internal emails. AutoForwarding emails can have many legitimate purposes in Exchange Online. Examples of these are:

- Employee that has left or retired and emails to their account are AutoForwarded to another user at the company so that important emails can be managed properly.
- A person or LOB app may send emails to a certain mailbox which then AutoForwards to another destination for processing by an app or other personnel.
- A mailbox that represents a board member that may want to receive their email at a personal email address in order to correspond with the sender.

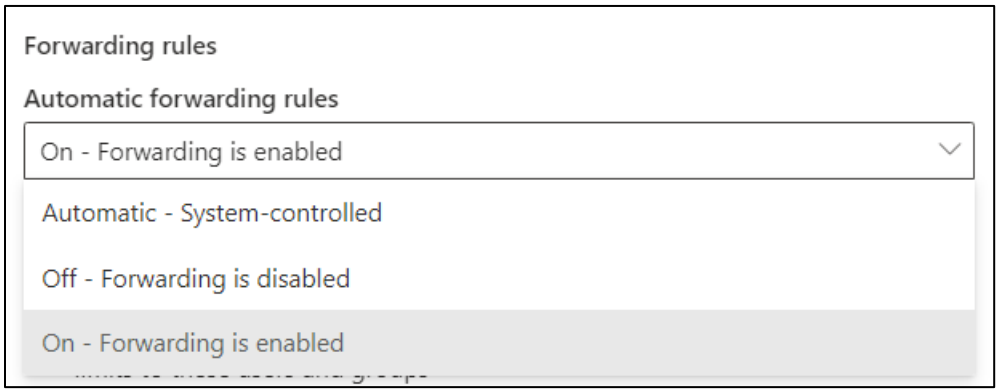
On the opposite side of legitimate reasons for AutoForwarding, there are also malicious reasons for AutoForwarded messages:

- An employee sending all emails to their personal Gmail or Yahoo accounts for 'safe keeping'.
- A hacker who has hijacked a mailbox and sending emails externally.
- … and so on …

In order to mitigate the malicious side of things, Microsoft introduced a feature to handle AutoForwarded emails in an environment. Where can we find these feature in Office 365? It is located in the **Microsoft Defender portal > Policies & Rules > Threat Policies > AntiSpam > OutBound Spam**:



Editing the Default Outbound Spam policy, we see a few options for the AutoForward feature in our tenant:



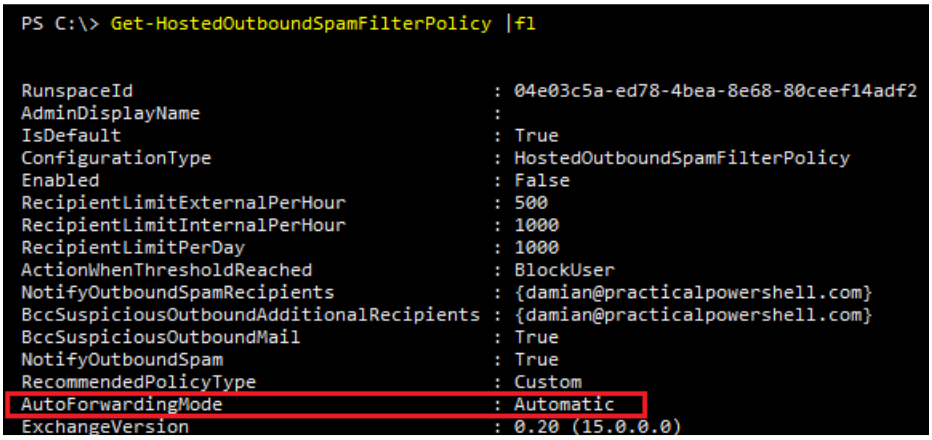
The three settings are:

Automatic: By now, this setting will turn off AutoForwarding in a tenant. If AutoForwarding is desired, then choose 'On'.
Off: Blocks AutoForwarding like Automatic does.
On: Allows AutoForwarding in a tenant.

PowerShell
We can see this setting exposed in Exchange Online's Hosted Outbound Spam Policy:

```
Get-HostedOutboundSpamFilterPolicy |Fl
```

Which shows us this (default setting):



This property currently accepted Automatic, On or Off as valid values. Changing this requires the use of the Set-HostedOutboundSpamFilterPolicy cmdlet like so:

```
Set-HostedOutboundSpamFilterPolicy -Identity Default -AutoForwardingMode Off
Set-HostedOutboundSpamFilterPolicy -Identity Default -AutoForwardingMode On
Set-HostedOutboundSpamFilterPolicy -Identity Default -AutoForwardingMode Automatic
```

No feedback is provided with the change.

Plus Addressing

Recently Microsoft added support for what is known as "+" email addresses or dynamic email addresses. This feature was the number one rated feature in User Voice and thus Microsoft brought the idea to fruition. Using plus addresses is a common way to help separate out and organize email traffic based on what address an email was sent to. Exchange Online

supports email addresses that contain a plus sign, but this is not Plus Addressing. The logic for Exchange Online is something like this:

Normal, supported email addresses:

```
<alias>@<domain>
```

Plus Addresses look like this:

```
<alias>+<tag>@<domain>
```

PowerShell
As this feature is off by default, we need to turn this on like so:

```
Set-OrganizationConfig -AllowPlusAddressInRecipients $True
```

This will enable the Plus Addressing feature. With this enabled, if you had signed up for a newsletter or provided your email address for marketing material in a form like this <alias>+<tag>@domain.com, then when an email is received, Microsoft will resolve it like so:

<alias>@domain.com

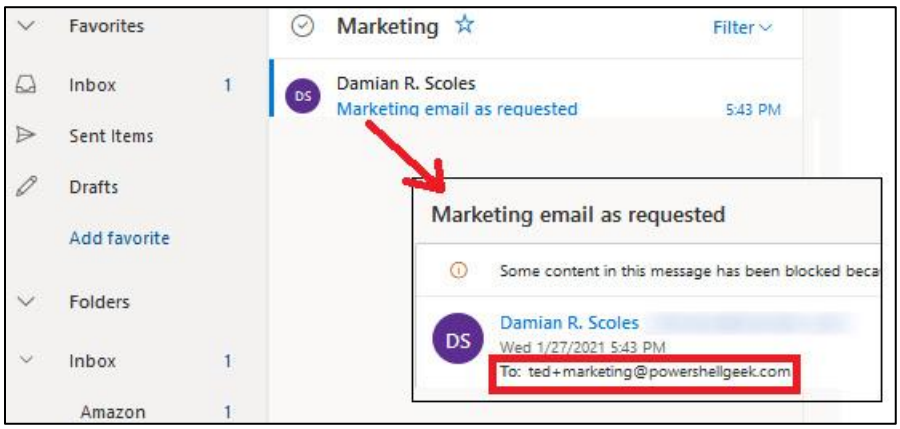
Example
Let's say we have a user who only has email aliases without '+' addresses would like to sign up for some monthly material but would like to keep these emails segregated into different folders. He wants to do this with Plus Addresses of Newsletter, Marketing and Amazon. He then signs up for these services providing these email addresses:

```
Ted+newsletter@powershellgeek.com
Ted+marketing@powershellgeek.com
Ted+amazon@powershellgeek.com
```

We have two ways to enable this to work for our end user. We can:

- Add these email addresses as aliases to his mailbox
- Enable Plus Addressing for Exchange Online

The far easier option is to choose the Plus Addressing option as it allows for dynamic options, less administrative overhead and a 'self-servicing' model. Now if an email comes in and Ted has a rule to process it, the email will go to the appropriate folder:



Notice the Plus Address in the To: field. This is a valid email address that Exchange Online has now accepted and used in rule processing for this mailbox.

Caveats

There are caveats to this, so if you already have Plus Addresses in your environment, prior to turning this feature on, it could cause issues as the email addresses will be viewed differently once that switch it turned on. For example a user has this alias pre-change:

```
Ted+technews@powershellgeek.com
```

Exchange Online will interpret this as:

```
Ted+technews@powershellgeek.com or <alias>@<domain>
```

When we turn on Plus Addressing, Exchange Online will interpret this address like so:

```
Ted@powershellgeek.com or <alias>+<tag>@<domain>
```

If this user does not have an alias of 'Ted@powershellgeek.com' the email will fail to be delivered. In the end, know your environment before making this change as it could have unseen side effects. You may want to verify no Plus Addresses are currently in use in your environment. PowerShell can help you determine this.

```
$Mailboxes = Get-Mailbox -ResultSize Unlimited
$PlusCount = 0
Foreach ($Mailbox in $Mailboxes){
$EmailAddresses = $Mailbox.EmailAddresses
If ($EmailAddresses -like '*+*') {
$PlusCount++
}
}
```

An environment with zero Plus Addressing in place would have a $PlusCount variable value of zero as none would be found. Otherwise the variable $PlusCount would increment based on the number of aliases found. If some were found, then further analysis may be required to determine who has a Plus Address.

```
Foreach ($Mailbox in $Mailboxes) {
$EmailAddresses = $Mailbox.EmailAddresses
$DisplayName = $Mailbox.DisplayName
If ($EmailAddresses -like '*+*') {
Foreach ($EmailAddress in $EmailAddresses) {
If ($EmailAddress -like '*+*') {
Write-Host "$DisplayName , $EmailAddress"
}
}
}
}
```

Feedback, when a matching alias is found, looks like this:

```
Frank Stein , smtp:frank+newsletter@powershellgeek.com
```

As we can see, a matching email address was found.

Cloud Voicemail

As stated previously in the chapter, with the removal of Unified Messaging from Exchange 2019, those with on-premises Exchange Servers need to decide if they want to continue using Exchange 2016 Unified Messaging Servers or us Cloud Voicemail. The previous chapter covered the ins and outs of using PowerShell to manage Exchange 2016 Unified Messaging servers. Below we will quickly review Option 2, Cloud Voicemail.

Requirements

In order to use Cloud Voicemail, we need to meet some requirements which depend on what is already in place.

- Skype for Business 2019
- Hybrid configuration
- One licensed Teams user
- Office 365 Tenant

Setup

- Met prerequisites
- Configure Hybrid connectivity
- Configure Cloud Voicemail - on Skype Edge server
- Configure Hosted Voicemail policy
- Assign Hosted Voicemail policy to users
- Enable user for Cloud Voicemail

PowerShell Cmdlet Sets

New-CsHostingProvider - Configure a new Hosting Provider
Set-CsHostedVoicemailPolicy - Configure Hosted Voicemail Policy
Get-CsUser - Assign User Voicemail Policy
Set-CsUser - Enable user for Cloud Voicemail

Documentation / Further Reading

Below are some resources that are available to assist in migrating to Cloud Voicemail:

Planning Guide for Cloud Voicemail: https://docs.microsoft.com/en-us/skypeforbusiness/hybrid/plan-cloudvoicemail
Set up Cloud Voicemail: https://docs.microsoft.com/en-us/microsoftteams/set-up-phone-system-voicemail
Skype 2019 Integration: https://docs.microsoft.com/en-us/skypeforbusiness/hybrid/plan-um-migration